5/17/05

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04K 1/00 — A2

(11) International Publication Number: WO 99/08411

(43) International Publication Date: 18 February 1999 (18.02.99)

(21) International Application Number: PCT/IL98/00369

(22) International Filing Date: 6 August 1998 (06.08.98)

(30) Priority Data:

| | | |
|---|---|---|
| 121499 | 8 August 1997 (08.08.97) | IL |
| 121500 | 8 August 1997 (08.08.97) | IL |
| 124705 | 1 June 1998 (01.06.98) | IL |

(71)(72) Applicant and Inventor: STIEBEL, Jonathan [US/IL]; Goldberg Street 7/10, 76283 Rechovot (IL).

(74) Agent: FRIEDMAN, Mark, M.; Beit Samueloff, Haomanim Street 7, 67897 Tel Aviv (IL).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

**Published**

*Without international search report and to be republished upon receipt of that report.*

(54) Title: NEW OPERATION FOR KEY INSERTION WITH FOLDING

(57) Abstract

MultiDES based systems with bit–slice implementation, one embodiment of the method of the present invention, is a new cipher based on a modification of bit–slice implementation of DES. Therein, the exclusive–or is replaced within the F function with a form of multiplication. Thus, every simultaneous encryption depends in all of the bits of input into the s–box on every other parallel encryption. Any invertable group operation could be used in place of multiplication. The principle requirement is that every input bit will influence every output bit. The operation need not be easily invertable, for example, common multiplication using exclusive–or to fold the upper and lower halves of the result yields a strong candidate. The method of the present invention uses a careful form of folding so that the inputs to any s–box depend on at least half of the input bits. MultiDES based systems with bit–slice implementation are particularly preferred, one embodiment of the method of the present invention. The recommended key schedule for Feistel and other blocks ciphers uses the block cipher to cause complete mixing of the key bits and pseudo–random expansion into conveniently sized subkeys. A subkey chaining mode for influencing future encryptions of block ciphers in place of cipher block chaining mode is proposed. A Feistel structure allowing for further extension of block length for subkey chaining output is proposed.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| AL | Albania | ES | Spain | LS | Lesotho | SI | Slovenia |
| AM | Armenia | FI | Finland | LT | Lithuania | SK | Slovakia |
| AT | Austria | FR | France | LU | Luxembourg | SN | Senegal |
| AU | Australia | GA | Gabon | LV | Latvia | SZ | Swaziland |
| AZ | Azerbaijan | GB | United Kingdom | MC | Monaco | TD | Chad |
| BA | Bosnia and Herzegovina | GE | Georgia | MD | Republic of Moldova | TG | Togo |
| BB | Barbados | GH | Ghana | MG | Madagascar | TJ | Tajikistan |
| BE | Belgium | GN | Guinea | MK | The former Yugoslav Republic of Macedonia | TM | Turkmenistan |
| BF | Burkina Faso | GR | Greece | | | TR | Turkey |
| BG | Bulgaria | HU | Hungary | ML | Mali | TT | Trinidad and Tobago |
| BJ | Benin | IE | Ireland | MN | Mongolia | UA | Ukraine |
| BR | Brazil | IL | Israel | MR | Mauritania | UG | Uganda |
| BY | Belarus | IS | Iceland | MW | Malawi | US | United States of America |
| CA | Canada | IT | Italy | MX | Mexico | UZ | Uzbekistan |
| CF | Central African Republic | JP | Japan | NE | Niger | VN | Viet Nam |
| CG | Congo | KE | Kenya | NL | Netherlands | YU | Yugoslavia |
| CH | Switzerland | KG | Kyrgyzstan | NO | Norway | ZW | Zimbabwe |
| CI | Côte d'Ivoire | KP | Democratic People's Republic of Korea | NZ | New Zealand | | |
| CM | Cameroon | | | PL | Poland | | |
| CN | China | KR | Republic of Korea | PT | Portugal | | |
| CU | Cuba | KZ | Kazakstan | RO | Romania | | |
| CZ | Czech Republic | LC | Saint Lucia | RU | Russian Federation | | |
| DE | Germany | LI | Liechtenstein | SD | Sudan | | |
| DK | Denmark | LK | Sri Lanka | SE | Sweden | | |
| EE | Estonia | LR | Liberia | SG | Singapore | | |

NEW OPERATION FOR KEY INSERTION WITH FOLDING

Background: FIELD OF INVENTION

This invention relates to using a form of multiplication as the key insertion operation and related folding methodologies useful to form a shorter input length keyed hash function.

Bit-slice methodology is used in one of the preferred embodiments of the method of the present invention.

## BACKGROUND: PRIOR ART

The classic approach to cryptographic hashing has been proposed by Ron Rivest of MIT in a function called MD5 (Message Digest 5, or perhaps Merkle-Damgaard). A theoretical criticism was brought in the conference called Europcrypt '96 by James Massey in a talk, "The difficulty with difficulty." Massey contends that the a function of similar complexity to MD5 will invert the function. This criticism holds for all non-keyed proposed one way functions.

It being understood that where reference is made in an embodiment of the present invention to any cryptographic primitive, especially MD5, and derivatives thereof, MultiDES based systems, subkeychaining mode, as well as MultiDES based systems with bit-slice preferably and optionally are employed. A version of MD5-MAC is referred to in Menezes, van Oorshot, Vanstone, "Handbook of Applied Cryptography," CRC Press, New York, 1996. Entropy, cipher-block-chaining, probabilistically checking for correctness and other cryptographic terms are defined in Menezes, et al. Descriptions found for MD5-MAC are not easily understood, nor is the rationale for the particular construction easily known.

The inventor of the present invention has proven Massey's conjecture for a simplified version of MD2 (a hash function allegedly by Ron Rivest). Viewed as a whole, all but one step of MD2 is an involution. Thus, the inverse function is not just of the same *complexity* as MD2, but is identically the same function. This is a very undersirable property for a hash function.

There exist modifications of MD5 which allow for keyed hashing. However MD5 is not deeply understood and has not undergone extensive analysis. Hans Dobertin has found some collisions (two

inputs yielding the same output) in the hash function MD4, forcing the publication of the additional complexity of MD5.

Another prior art approach is to use classical block symmetric algorithms for hashing.

CAST is obviously different from the method of present invention in that it uses expansion-based s-boxes. Thus, fewer bits (8 bits) yield 32-bit outputs for the s-boxes. Use of CAST relies on esoteric properties of bent functions, it is difficult for many people to understand their s-box design principles so as to be able to place the necessary amount of trust in them.

Whitfield Diffie commented, Eurocrypt '98, to the inventor of the present invention that it would take him, one of the founders of public-key cryptography, a full year to understand s-box design. Thus, the cryptographic community finds significant obstacles to understanding and verifying different s-box designs.

IDEA U.S. patent #5,214,703 in its current form does not have a block length of 128 bits. It is different from the method of present invention because a preferred embodiment of the method of present invention maintains the overall Feistel structure of DES, changing mainly the key-insertion and scheduling operations. The operation shown "(x)" in section 3.4.1 (p.34) (On the design and security of block ciphers by X. Lai and J. Massey) differs in content and purpose from the method of the current invention. In content the operations are performed at once on four sets of inputs and are strictly single algebraic group operations, and in purpose no extension of block length is achieved. Outside reviews of IDEA are not widely available due to its relative newness.

Another application of the method of the present invention is in a Message Authentication Code (MAC). An approach to accomplish a MAC was brought in the Bracthl U.S. Patent # 4,908,861 differs from our construction in that we provide folding within the round function and Bracthl does so only on the entire DES encryption. The method of the present invention's construction provides more though mixing by using a form of multiplication within the round. It is unknown how much cryptanalysis the MDC2 and MDC4 modes have withstood.

An extremely different approach to hashing could be constructed using RC5, again by Ron Rivest. As he is the author of MD5 above, he favors MD5 for that application. RC5 is obviously different from the method of the present invention in that it uses data-dependent rotations as its principle

operation. This operation may have significant drawbacks. In a recent attack on RC5 in Eurocrypt '98, weaknesses are shown in a slightly modified version.

The inventor of the present invention had a part in the earlier stages of the mentioned attack on RC5 during his attacks on data-dependent rotations as a cryptographic consultant.

RSA U.S. patent #4,405,829 is different from the method of the present invention because the method of the present invention uses the same key for encryption and decryption. The system of the present invention is based on classical (i.e. Shannon 1949) confusion and diffusion rather than pure algebraic structures. RSA, due to its algebraic structure, has the multiplicative property, that is encryption_of_ *a* times encryption_of _*b*= encryption of a product *a*b*. Thus, RSA is not appropriate for use as a hash function or to encrypt arbitrary user-supplied data. (For example, see Coppersmith, Eurocrypt '96 for some attacks on RSA.)

An attempt to use DES U.S. patent #3,962,539 in its current unmodified form would not be appropriate because of the tiny key-block lengths. DES is different from the method of the present invention because the key insertion operation has been changed, the key schedule revised, and a methodology of folding introduced to yield larger block and key sizes. The description of the method of the present invention hereby incorporates specifically the patent 3,962,539 by reference to define the terms s-box, E Expansion, key schedule, P permutation. (See tables II-IV.) Descriptions of prior art DES implicitly refer thereto. The reader is referred also to FIPS PUB 46-1, Data Encryption Standard and FIPS PUB 81, DES Modes of Operation. It being understood that where reference is made to DES and derivatives thereof as part of an embodiment of the present invention, MultiDES based systems as well as MultiDES based systems with bit-slice preferably and optionally are employed.

The U.S. patent of Feistel #3,798,359 was filed in 1971, DES was based on the Feistel structure. The reader may refer to a substantially equivalent description in [BiSh93, appendix A].

A bit-slice fast-parallel bit-wise vector implementation of DES is referred to in Biham, E., "A fast new DES implementation in software," Proceedings of Fast Software Encryption Workshop, Springer-Verlag, January 1997.

On page 8, Biham states that: "We can use this fast code to design a new, even faster, and more secure cipher, which we call WDES. We convert the code by removing IP, FP, and changing the EPS

operations (S boxes followed by P followed by E, as used in this implementation) into S boxes from 8 to 64 bits. These S boxes can be much better than the original, since each S box affects all of the bits of all of the S boxes in the next round (rather than one bit in only six S boxes)."

A bit-slice implementation relies on bit-wise attribute used for key infusion inside the F function. It requires redesigning the substitution boxes of DES in the form of logic gates. Biham implementation of the logic gates was appropriate for exactly 64-bit machines. The method of the present invention is appropriate particularly for 32-bit machines The Biham method for WDES was appropriate for exactly 64-bit output s-boxes each. The method of the present invention is appropriate particularly for 32-bit machines, thus 32-bit output s-box.

Trying to use bit-slice DES for hashing would immediately fail because fundamentally, it is just a collection of DES operations operating in parallel without interaction between them. Bit-slice DES is different from MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention, because the key insertion operation has changed. Thus MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention, does not share the equivalence between bit-slice DES and DES.

The structure of DOS directory file entries is referred to in "PC intern: The Encyclopedia of System Programming," Tischer and Jennirch, Abacus, 1996. The PC Intern document defines a variety of terms of the art including "file handle", "opening a file", "FAT", "hard drive", "hard drive serial number", "sector number", "number of read/write heads" and "cluster." The term "file" is intended to include a directory or a directory tree. An "attribute byte" is a byte within the directory entry of a file as defined in PC Intern.

Differential Cryptanalysis is a methodology for attacking ciphers. Biham and Shamir teach against a preferred embodiment of the method of the present invention. Biham, E. and Shamir, A., Differential Cryptanalysis of the Data Encryption Standard, Springer-Verlag, New York, 1993 in section 4.5.3.1 of chapter 4 states that "If we replace the exclusive or operation within the F function by an addition operation we get a much weaker cryptosystem." The term "F function" and other conventional DES terms such as subkey as well as how to decrypt using DES are explained in Biham and Shamir's Appendix A, "Description of DES" and/or in the Glossary of the Biham and Shamir publication. On page 14, Biham and Shamir remark that "to simplify the mathematical analysis of our

attacks, we assume that all subkeys are independent. Attacks on DES with dependent subkeys were experimentally shown to have the same probability of success, but the theoretical analysis of the probability is much harder." The method described in the theory of the present invention grapples with issues where carry is present such as in addition or multiplication based operations. In the theory of present invention, Biham-Shamir assumption implying that success of an attack is independent of particular key chosen is shown not to apply.

The disclosures of all publications mentioned in the specification and of the publications cited therein are hereby incorporated by reference.

**PRIOR ART: DATA ENCRYPTION STANDARD, FEISTEL STRUCTURE**

BACKGROUND: DES (Data Encryption Standard) was developed by IBM with advice from the NSA (National Security Agency) of the United States of America. The NSA also made modifications to the S-boxes. DES is one of the most widely employed encryption algorithms. The Data Encryption Standard is built as spelled out by the above referenced Biham-Shamir publication. Its speed is 12,000 bytes per second on a Pentium 120 Mhz machine.

The standard implementation is as follows:

*Inputs: 64-bit key and 64-bit plain-text. (Reference is made to figures 14-15.)*

***KEY SCHEDULE***

*The 64-bit key enters an initial permutation, which results in a 56-bit key being used. Then on each round, the 56-bits are split into 28-bit halves. Each half is left circular shifted 1 or 2 bits depending on the round. A compression permutation selects 48 out of 56 bits for use in a round.*

***FEISTEL STRUCTURE AND DES ROUND***

*The 64-bit plain text also undergoes an initial premutation. Then it is split into 32-bit halves. On each round, the right half undergoes an expansion permutation which results in 48 bits -- 16 of the 32-bits are repeated. The key bits from the compression permutation and the input bits from the expansion permuation are applied the function exclusive-or, and then split into 8 6-bit units, each an input to one of 8 S-boxes. The S-boxes consist of 4 rows by 16 columns of values from 0 through 15. The outer 2 bits of a 6-bit input determine the row (hex: 0..3) and the middle 4 bits determine the column (hex: 0..F). The output is the value contained in the row, column of the S-box.*

*The 8 S-boxes each yield 4 bits for a total of 32. These bits undergo a P-box permutation, which mixes these bits. Finally, the bits are applied the function exclusive-or with the left half of the round input bits. Then the left half becomes the right half, and the applied the exclusive-or function result becomes the left half, for the start of the next round. At the end of the final round, the right half remains as is, and the applied the function exclusive-or result replaces the left half.*

## OBJECTS AND ADVANTAGES

Accordingly, several objects and advantages of the preferred embodiment of the method of the present invention are speed, simplicity of design, cryptographic strength and flexibility in key-block lengths. Herein the prior art will be assumed to be Triple-DES (DES used three times with two distinct keys of 56 bits each).

One advantage and object of the method of the present invention is superior speed relative to the prior art. Due to the increased resistance to cryptanalysis and large block size, the method of the present invention achieves better security than triple DES in fewer than 16 rounds.

Another advantage and object of the method of the present invention is additional larger block size. This allows for hashing, stream cipher applications, and resistance to birthday attacks wherein the same input/output pairs indicate a correspondence within the underlying scheme. In addition to the required key-block sizes, the method of the present invention, optionally and preferably, provides 192-192 key-block size and 256-256 key-block size. The speed per byte encrypted is faster on the larger block sizes. For encrypting large amounts of data or data with significant local structure, a large block size such as 256 bits is necessary for security.

Another advantage and object is resistance to differential cryptanalysis. The use of the multiplication operation in combination with the complex folding causes classical methods of differential cryptanalysis difficulty. It is important in any new system to address this approach.

Another advantage and object is key size and flexibility. The method of the present invention provides for a variable key size ranging from 40 bits to 256 bits for its 64 bit block size to 256 bit block size modes. Each key bit which is used has an impact on the resulting encryption.

Another advantage and object is flexibility in key setup time required. An embodiment of the method of the present invention generates new keys while it encrypts. Thus, key setup takes just one

encryption time. In another embodiment of the method of the present invention, the key schedule performs any user defined plurality of rounds between sampling material.

User choice of 4, 8, 16, or 32 are recommended due to studied properties of the Feistel structure. For example, after 4 rounds, the Feistel structure is "complete." Completeness is that each input bit has the opportunity to influence each specific output bit. As another example, after 16 rounds, the Feistel structure has executed four sets of four rounds each. Thus, should an F function be chosen which is substantially similarly to that used in prior art DES, yet failing to have all the 32 output bits in each round depend on each input bit, substantially 16 rounds provides completeness under less demanding assumptions on the properties of the F function.

Thus, for applications which need to rapidly change keys, a preferred embodiment of the method of the present invention does so. For applications which require security against key search, another preferred embodiment of the method of the present invention does so.

Another advantage and object is reusability. The method of the present invention uses the encryption algorithm to accomplish rapid and secure key scheduling. The method of the present invention uses the well-tested E Expansion, S-Boxes, P Permutation and Feistel structure of prior art DES patent #3,962,539 . The method of the present invention uses commonly available multiplication. All of the constant values present in the preferred embodiment of the method of the present invention are available in implementations of prior art DES. Using widely available constant values increases the confidence level of potential users of the method of the present invention.

Another advantage and object is compact implementation. The method of the present invention in the preferred embodiment for every specific key-block size less than a thousand bits each and mentioned herein has been implemented in ANSI C in less than 3/4 of the size of a comparable DES implementation.

Another advantage and object is simplicity of design. The method of the present invention changes the key insertion operation within the F-function to include multiplication. The folding which becomes possible thereby enables arbitrarily long block sizes using a simple and regular construction.

Another advantage and object is that the method of the present invention can control a microprocessor to create the output of a hashing algorithm. The embodiment of the method of the

present invention with 256-bit block size can be used as a keyed or non-keyed hashing function in place of MD5. One preferred embodiment of the method of the present invention takes the output upper and lower 128-bits of the output to be arguments to the function of exclusive-or to yield a single 128-bit output. Another preferred embodiment of the method of the present invention uses an exclusive-or of the input plain text with the output cipher text to yield a 256-bit block output.

Another preferred embodiment of the method of the present invention allows the round input to the new F function to be dependent on substantially more than half of the bits of the given block size. Another preferred embodiment of the method of the present invention allows the round output of the new F function to influence substantially more than half of the bits of the given block size. These two preferred embodiments differ substantially from the classic Feistel structure referred to herein.

Another preferred embodiment of the method of the present invention is to use an exclusive-or of the round plain text derived input with the round plain text derived output to yield the new plain text derived output.

Even if the key were to be published, it would still be hard to invert.

The advantages would include:

1. Smaller input block size of 256 instead of 512.
   This is often more convenient for small data items such as passwords.
2. Natural method for keying built into the cipher.
3. Easier to understand and clearer design principles.

Another advantage and object is ability to define a new mode of operation which derives from execution of the cipher in key-generation mode and using the newly generated subkeys for future encryptions. The key-generation mode also produces cipher text of the desired plain text.

Another advantage and object is handling high bandwidth or highly structured inputs whose structure often remains apparent when using small block size ciphers. The large block size and effective mixing which is apparent after a mere four rounds of the cipher provide protection against matching based "birthday" attacks as well as scrambling the local patterns better than just CBC mode. A fast gate-based implementation is available for large block sizes.

Another advantage and object of the method of the present invention is that it does not exhibit known weak keys, complementation properties, or have self-complementing keys.

Another advantage and object of the method of the present invention is that even in a simplified method, the round-dependent masks would cause weak subkeys to be round-dependent. Such a restriction greatly reduces the usefulness of such a weak subkey to attack the system.

An advantage and object of a method of the preferred embodiment is application to ATM Networks. These networks have a high bandwidth. Thus fast algorithms processing a large block size are advantageous here. The increased block size, speed and resistance to cryptanalysis with respect to Triple DES gives **TMD** an advantage for this application.

An advantage and object of a method of the preferred embodiment is application in cipher feedback mode or cipher block chaining mode or subkey-generation chaining mode to yield a stream cipher.

An advantage and object of a method of the preferred embodiment employs cipher-block-chaining (hereinafter CBC) with a random initialization vector (hereinafter IV) generated using counter mode and a secret key to yield ciphertext which yields no computational information about the plaintext.

An advantage and object of a method of the preferred embodiment can be employed after applying the function of exclusive-or to a plain text with a key stream, and followed by applying the function of exclusive-or to a cipher text with a key stream.

A preferred embodiment of the method of the present invention, **TMD**, is a CBC based message authentication code (hereinafter MAC). Apply the CBC mode to the text after padding (if necessary). Decrypt the final result using another secret key. This is the MAC result. There is no need for a random IV in this case. Some DES based MAC's can be attacked in $2^{32}$ time due to their small key–block size. However, here a block size of a minimum of 128 bits would be suitable, making this approach far superior to current MAC's using DES.

An advantage and object of a method of the preferred embodiment is application to High Definition Television, Satellite and Voice Applications. A large block cipher combined with a rapid execution time provide **TMD** with advantages for this application. The dependence on highly defined and structured data means that reliance on cipher-block-chaining with block ciphers of a short length is not recommended. One can learn the plain text exclusive-or from a repeating of cipher blocks.

Another advantage and application being in accordance with another preferred embodiment of the present invention is a system for protecting confidentiality of information written on a notebook computer, the system including: an automatic file-by-file information protector operative to protect a plurality of files on an automatic file-by-file basis, the information protector including: a symmetric encryptor using a symmetric cryptosystem to encrypt an individual file, thereby to generate an encrypted individual file; and a notebook storage manager operative to store the encrypted individual file on a notebook computer.

Another object and application being in accordance with another preferred embodiment of the present invention is a system for protecting confidentiality of information written on a hard disk, the system including: a symmetric file encryptor using a first symmetric cryptosystem to encrypt a file having a selectably known file key; and a symmetric file key encryptor operative to encrypt the selectably known file key using a second symmetric cryptosystem and a selectably known master key derived from a selectably known pass phrase using a cryptographically strong hash function.

Further objects and applications of the present invention will become apparent from a consideration of the drawings and ensuing description.

**DESCRIPTION of DRAWINGS**

The present invention will be understood and appreciated from the following detailed description, taken in conjunction with the drawings in which:

Figure 1 is an exemplary illustration of a preferred method explaining how to make and use an encryption and decryption portion of the method of the present invention;

Figure 2 is an exemplary illustration of a preferred method explaining how to make and use the key insertion portion of the method of the present invention; preferably, the form of multiplication chosen will be common product of the two arguments plus exclusive or of the arguments;

Figure 3 is an exemplary illustration of a preferred method explaining how to make and use an operation on two inputs yielding a double-sized result portion of the method of the current invention; preferably, fold half of the double-sized result into a companion execution of the method.

Figure 4 is an exemplary illustration of a preferred embodiment which explains how to make and use the key schedule portion of the method of the present invention;

Figure 5 is an exemplary illustration of a preferred embodiment which explains how to make and use the substantial key-block size portion of the machine of the present invention;

Figure 6 is an exemplary illustration of an optional embodiment which explains how to make and use an improved key schedule portion of the method of the present invention; preferably, including feeding the full 64 key bits per block into a rearranged PC2 from prior art DES;

Figure 7 is an exemplary illustration of a preferred embodiment which explains how to make and use the form of multiplication portion of the method of the present invention;

Figure 8 is an exemplary illustration of a preferred embodiment which explains how to make and use a permutation to span multiple blocks portion of the method of the present invention;

Figure 9 is an exemplary illustration of a preferred embodiment which explains how to make and use a circuit-based logic-gate implementation of the machine of the present invention;

Figure 10 is an exemplary illustration of a alternative embodiment which explains how to make and use masks derived from DES s-box entries; table I, a Key Selection Permutation Table in an improved key schedule designates which master key bits will be selected for each round subkey;

Figure 11 (top) is an exemplary illustration of a preferred embodiment which explains how to make and use inputs of master-key and predetermined initial keys to yield master key derived subkeys;

Figure 11 (middle) teaches to make and use inputs of master-key and predetermined initial keys to yield master key derived subkeys; these subkeys are used to encrypt in key-generations mode a plain text, which in turn generates additional subkeys as well as a cipher text;

Figure 11 (bottom) teaches how to make and use subkey-feedback-mode;

Figure 12 is an exemplary illustration of a preferred embodiment of the method of the present invention which explains how to make and use the encryption and decryption portion of the method of the present invention. It differs from figure 1 in that it recites fewer optional elements;

Figure 13 is an exemplary illustration of a preferred embodiment which explains how to make and use the key schedule portion of the method of the present invention; it differs from figure 4 in that it recites fewer elements and generalizes to non-Feistel methods;

Figure 14 is an exemplary illustration of a preferred embodiment which explains how to make and use an internal round function portion of the method of the present invention;

Figure 15 is an exemplary illustration of a preferred embodiment which explains how to make and use a Feistel structure for Multi-DES portion of the method of the present invention;

Figure 16 is an exemplary illustration of a preferred embodiment which explains how to make and use a particular form of multiplication portion of the method of the present invention;

Figure 17 is an exemplary illustration of a preferred embodiment which explains how to make and use an example round function for **TMD** using two MultiDES encryptions in tandem;

Figure 18 is an exemplary illustration of a preferred embodiment which explains how to make and use an example round function **TMD** using three MultiDES encryptions in tandem;

Figure 19 is an exemplary illustration of a preferred embodiment which explains how to make and use an example round function **TMD** using four MultiDES encryptions in tandem;

Figure 20 is a simplified flowchart illustration of a preferred method for protecting data on a notebook computer;

Figure 21 is a simplified flowchart illustration of a preferred method for protecting confidentiality of information written on notebook computer, the method being constructed and operative in accordance with a preferred embodiment of the present invention;

Figure 22 is a simplified flowchart illustration of a use of a slightly modified MD5-MAC message authentication code method constructed and operative in accordance with a preferred embodiment of the present invention;

Figure 23 is a simplified flowchart illustration of a preferred method for generation of file keys forming a part of the method of figure 22, using contents of DOS directory entries as plain texts and keys to generate a file key;

Figure 24 is a simplified flowchart illustration of preferred method for performing an encryption of a file using the method of figure 23 to generate file keys and the output of the method of figure 22 to protect the file key;

Figure 25 is a simplified flowchart illustration of preferred method for performing an encryption of a file on a sector by sector basis using unique information based on the location on the particular hard disk and cipher-block-chaining within the sector; and

Figure 26 is a simplified flowchart illustration of preferred method for performing the method of figure 25 wherein the encryption is fast parallel bit-wise vector implementation of DES with a form of

multiplication substituted for exclusive or when combining the subkey with the plaintext derived input.

Figure 27 is a simplified flowchart illustration of a DES encryption method constructed and operative in accordance with a preferred embodiment of the present invention;

Figure 28 is a simplified flowchart illustration of a first preferred method for performing an n'th DES round forming part of the method of figure 27, using addition to combine subkey with plain text derived input;

Figure 29 is a simplified flowchart illustration of a second preferred method for performing an n'th DES round forming part of the method of figure 27;

Figure 30 is a simplified flowchart illustration of a modification of figure 2 in which first and second permutations and mapping are employed to perform the DES round.

Figure 31 is a simplified flowchart illustration of a third preferred method for performing an n'th DES round forming part of the method of figure 27;

Figure 32 is a simplified flowchart illustration of a DES encryption method constructed and operative in accordance with another preferred embodiment of the present invention;

Figure 33 is a simplified flowchart illustration of a fourth preferred method for performing an n'th DES round forming part of the method of figure 32, using multiplication to combine subkey with plain text derived input;

Figure 34 is a simplified flowchart illustration of a fifth preferred method for performing an n'th DES round forming part of the method of figure 32;

Figure 35 is a simplified flowchart illustration of a modification of figure 33 in which first and second permutations and mapping are employed to perform the DES round;

Figure 36 is a simplified flowchart illustration of a sixth preferred method for performing an n'th DES round forming part of the method of figure 32;

Attached herewith are the following appendices which aid in the understanding and appreciation of one preferred embodiment of the invention shown and described herein:

A separate section entitled *Theory of present invention* has been printed by itself, yet is to be read as an integral part of the disclosure. These findings build on research findings which indicate that replacing the exclusive-or function with an addition operation does not always yield a weaker

cryptosystem, contrary to the teachings of Biham and Shamir in Section 4.5.3.1 of Chapter 4 of the above-referenced Biham-Shamir publication.

**FORM OF MULTIPLICATION**

An advantage and object of employing a form of multiplication to accomplish key insertion being ability to demonstrate in a Theory of Operations section the strength of method of the present invention by attacking using differential cryptanalysis on a simplified version. The simplification is employing as a form of multiplication as common multiplication with carry discarded.

Attempts at using differential cryptanalysis with a ratio to cancel the key insertion face building difference distribution tables over all 16 to 32 bits at once due to interference with the P Permutation.

Thus, the preferred embodiment of the method of the present invention employs a form of multiplication in place of exclusive-or as the key insertion operation due to its better mixing and consequent resistance to cryptanalysis.

Another advantage and object of a preferred definition of multiplication is that it allows a pair of values to be blended whereby the upper half of one product has the exclusive-or function applied with the lower half of the companion product.

A form of multiplication can be selected from the group including:

(a) in the algebraic sense, i.e. any operation on two arguments yielding a third, e.g. elliptic curve

(b) common multiplication

(c) multiplication over a ring

(d) multiplication over a field (or nearly a field)

(e) multiplication over a Fermat or Mersenne field (or nearly a field)

(f) common multiplication, yielding an upper and a lower, linear combination thereof.

(g) common multiplication of $n$ inputs to yield a interum product, exclusive-or between subsets of those $n$ inputs to yield a mix, optionally concatenate distinct mixes to yield length equal to interum product, sum together interum product and (concanated) mix to yield a product.

(h) any of the above forms of multiplication on a plurality of arguments

(i) any of the above forms of multiplication where at least one argument is a constant

It being understood that operations described herein for brevity as a form of multiplication may have their implementation optimized to eliminate a common multiplication operation, representation

not withstanding it shall still be considered herein a form of multiplication. Clearly, any multiplication can be rewritten in the form of additions and shifts, yet it is still understood to be multiplication.

Reference is made to figure 2. The preferred embodiment of the present invention employs a form of multiplication to do key insertion. Form employed can be multiplication over a Fermat field such as $2^{16}+1$. Alternatively, the method of the present invention employs common multiplication with carry discarded as the form of multiplication. Optionally, a form of multiplication includes addition or multiplication of points on an elliptic curve. (See for example, Silverman, *Arithmetic of Elliptic Curves*, 1986, Springer-Verlag.) Optionally, any operation in which the "*" operator is reused in object-oriented languages such as Ada or C++ is a form of multiplication.

In a preferred embodiment of the method of the present invention, a form of multiplication can be understood as the operation on *a, b* defined by *a*b+*(*a* exclusive-or *b*), where * is common multiplication, + is common addition, and exclusive-or is common exclusive-or. This definition of a form of multiplication is novel and non-obvious. The term "product" is defined herein to refer typically to this form of multiplication, wherein the two variables used for illustrative purposes only could be a plurality of variables such as *a*b*c+*(*a* exclusive-or *b exclusive-or c*). Alternatively, the term "multiplication" in particular with a plurality of variables, could be defined as *a*b*c+(a ^b << 32 | b ^ c << 16 | c ^ a)*. Alternatively, the form of multiplication with a plurality of variables is defined as *a*b*c*d | (a^b^c << 48 | b^c^d << 32 | c^d^a << 16 | d^a^b)*. Following ANSI C conventions, "^" is exclusive-or, "|" is binary-or, and "<< x" means shift left x bits. Implicitly, the example herein assumes a word size of 16 bits, however, this is strictly exemplary as any word size could be suitably employed by dividing "x" by 16 and multiplying by the desired word size.

The term "resultant product" is therefore typically the result of such blending of a plurality of products. Looking at the resultant product, it substantially retains the benefits of modulo multiplication using the common definition of multiplication.

In another embodiment of the method of the present invention, a form of multiplication can be understood to be the operation on *a, b* defined by *a*b* wherein the variable *lower* is assigned the lower half of the product and the variable *upper* half of the product. The result shall be a linear combination of *upper*, and *lower* for example for constants *c1, c2*: *c1*upper+c2*lower*. Alternatively,

understand linear combination to be *c1*upper* exclusive-or *c2*lower*, as well as substantially similar constructions.

When present in a Feistel structure, a key-inserter may use any form of multiplication desired. The term "key-inserter" is not intended to be used where the form of multiplication is multiplication modulo 65537 to form a first product followed by addition modulo 65536 using that product followed by multiplication modulo 65537 to form a second product followed by addition modulo 65536 between the first and second products.

When employing a multiplier, an exception is treating a one or more input values distinctly. An non-exclusive indicator of exceptions are conditional constructions in programming languages. A logarithmic number of exceptions is a limited number of conditional constructions, for example, less than 16 for the field modulo 65537.

If the length of each integer is 8 bits and if multiplication over a ring is employed then the ring may, for example, be modulo 257 wherein 0 is considered to be -1. If the length of each integer is 16 bits and if multiplication over a ring is employed then the ring may, for example, be modulo 65537 wherein 0 is considered to be -1. If length of each integer is 32 bits and if multiplication over a ring is employed then modulus of the ring is typically slightly in excess of $2^{32}$.

Optionally, justification for broadening definition of a form of multiplication to include variant forms is due to mathematical fact that multiplication modulo 2^n+1 can be calculated in such a linear combination manner with subtraction used suitably to yield correct identity in linear combination.

Alternatively, a form of multiplication is understood to include multiplication over a ring.

Alternatively, a form of multiplication is understood in the algebraic sense thus an operation on two arguments yielding a third. Clearly, addition or exponentiation is understood in algebraic context to be a form of multiplication. Thus, language such as performing a round function employing a form of multiplication is understood to include employing common addition, addition with carry discarded, common multiplication and common multiplication with carry discarded, and others.

For example, employing a key insertion operation of a form of multiplication within DES, would describe add-DES wherein addition is substituted for exclusive-or in the round function. Use of non-symmetric operations such as subtraction, or division are herein considered to be a form of

multiplication. In an embodiment of the present invention, division over a field is accomplished by finding the multiplicative inverse, followed by standard multiplication over the field.

An advantage and object of the present invention is achieving a product which benefits from the long-range effects of carry present in multiplication together with the preservation of hamming weight independence provided by exclusive-or. An alternative form of multiplication in which zero is treated as negative one, is believed alternative to a preferred definition of the form of multiplication. Through experimental work, the alternative form of multiplication was shown to have so-called "weak keys". Keys with either a high or low Hamming weight would cause less satisfactory results using the alternative form. Preferably, the implementation of the above definition of product as a plurality, yields an additional novel and unobvious way of mixing values in companion executions of modified round functions.

An advantage and object of the present invention is to achieve as thorough mixing of distant bits as is possible in modulo multiplication.

Another advantage of the preferred embodiment of the present invention with a preferred definition of multiplication is that it allows a pair of such resultant products to be blended.

**FORM OF FOLDING**

A form of folding operates on a pair of double-length results of *a form of multiplication* to yield a single double-length result. A preferred embodiment of the present invention performs exclusive-or between the upper half of a first double-length result and the lower half of a second double length result to yield a first mix. Preferably, in addition exclusive-or is performed between the lower half of the first double-length result and the upper half of the second double length result to yield a second mix. Further preferably, concatenate the first mix to the second mix to yield a folded result.

Herein a form of folding includes performing at least one application of an element selected from the group consisting of a form of multiplication, a form of folding, and a form of blending.

Folding refers to a wide variety of operations available on computers, typically such operations are group operations and occassionally the operations are bit-wise. Folding a single-size portion into a companion execution implies application of a group operation between all of the single-size portions to be folded in, yielding a single size result. Typical folding can be addition or exclusive or.

Extended folding may involve pseudo-random expansion, perhaps employing a form of multiplication, in proximity to application of a group operation.

An object and advantage of the method of the present invention is to create a pair of outputs whose individual bits vary with each of a plurality of inputs.

Another object of the method of the present invention is to create a pair of outputs wherein individual bits vary with each bit of a plurality of inputs.

Another object and advantage of the method of the present invention is to extend block length of a cryptographic primative.

**FORM OF BLENDING**

A form of blending operates on a pair of double-length results of *a form of folding or a form of multiplication* to yield a single double-length result. A preferred embodiment of the present invention performs exclusive-or between the upper half of a first double-length result and the lower half of a second double length result to yield a first mix. Further perform exclusive-or between the lower half of the first double-length result and the upper half of the second double length result to yield a second mix. Further, concatenate the first mix to the second mix to yield a blended result.

Optionally and preferably, an embodiment of the present invention on a pair of double-length inputs performs a concatenation of the upper half of a double-length input and the lower half of the other double length input to yield a blended result.

An optionally and preferably, a form of blending operates on a *n* size input, yielding a single-size result. Optionally, a form of blending operating on a n size input, yielding a single-size output may employ a form of multiplication. Further optionally, the form of multiplication employed may be exclusive-or.

The result of preferably more than one distinct multiplication are combined in a blending operation. The blending operation on two arguments *a*, *b* returns 32-bit result wherein the upper half of the result is the lower half of *a*. The lower half of the result is the upper half of *b*. Preferably, the blending arguments *a* and *b* are chosen so that, when possible, *a* depends on different plain text derived inputs from *b*. Likewise, for every output of the multiplication it must appear exactly once on the left and once on the right arguments of the blend.

Thus, each s-box's input depends on a subkey-based pseudo-random expansion of half of the bits of the plain text derived input. Moreover, the bits are only 16 bits out of each 32 bit input block. Thus, the four s-boxes are two pseudo random expansions of half the input bits and two pseudo random expansions of the other half of the bits. For illustrative purposes, the embodiments feature 16 bit word-size, however any suitable word-size would be appropriate. The reader mentally divides 16, 32 where they appear in the text by 16 and multiply by the new word size.

Blending refers to a wide variety of operations performed with computational devices such as PC computers; typically such operations are permutations of bits. An example of an effective blending is selecting two groups of 16-bits out of distinct 32-bit quantities. Another example of blending would be selection of every fourth bit from four quantities. Another example of blending would include a plurality of group operations on the selected bits.

A combiner is a logic circuit which performs folding or blending as necessary. Combining is either folding or blending as necessary. Combining may also be forming a third permutation which is equivalent to a composition of two given permutations. Combining may also be forming a third mapping equivalent to a composition of two given mappings, for example s-boxes followed by e expansion or P permutation followed by E expansion.

Definitions appearing in this section are hereby extended to include defintions and implicit usage elsewhere in the text. For example, a form of multiplication, folding, and blending are understood to be broadened by descriptions elsewhere in this document and in figures.

## SUMMARY: METHOD AND MACHINE OF PRESENT INVENTION

The method of the present invention provides symmetric encryption using a form of multiplication to accomplish key insertion and allow for extension of block length.

Reference is made to figure 1. According to a preferred embodiment of the present invention there is provided a method for performing a round function of an iterated encryption for a plurality of 32-bit input blocks, the steps of the method being performed by a data processor, the method comprising the steps of: numbering the plurality of input blocks from "0" to "n" with an input block number; splitting each of the plurality of input blocks into an upper half and a lower half to produce plain text-derived input; combining plain text-derived input with a plurality of round-dependent subkeys according to a form of multiplication to form a blended product; applying a plurality of s-

boxes of the F function of a DES encryption algorithm to blended product; and applying the P permutation of the F function of a DES encryption algorithm to output of the s-boxes. An advantage and object is that each of the round output bits depends on at least half of the round input bits. Another advantage and object is enhancement of resistance to differential cryptanalysis. A number of failed attempts have been made in the prior art to extend the block length beyond 64-bits. The classic failure in the prior art is G-DES. (Documented and broken in [BiSh93].)

Reference is made to figure 2. A preferred embodiment of the machine of the present invention for encrypting comprising: a key-inserter which employs a form of multiplication for key insertion, whereby the block length of the encryption can be extended. Thus, localized visible structure is scrambled, particularly useful when data represents a picture or mobile set of pictures.

Reference is made to figure 3. Another preferred embodiment of the machine of the present invention wherein multiplication occurs in chunks at least as large as single bytes. An object and advantage is that the number will fit into common hardware registers. Another object and advantage is that the chunk may be chosen to apply over a Fermat field.

In the mentioned preferred embodiment of the machine of the present invention, further wherein the individual multiplications are carried out over a Fermat field. An object and advantage of multiplication over a field is that the result is known to be a permutation. Another advantage and object of multiplication over a field is that for any known output, there exists a key, which will transform the output to any desired input value. This property is referred to throughout this text hereinafter as a "group" operation. An operation with is substantially similar to this group operation will be called a "group-like" operation. An object and advantage of a group operation is that the output of the multiplication carries no information about the plain text input.

In the mentioned preferred embodiment of the machine of the present invention, wherein the form of multiplication in the key inserter comprises: common multiplication of arguments to yield a product, designating the upper and lower half of the product, combining the upper half with the lower half using exclusive-or to form a final product. An object and advantage of this embodiment is that the final product maintains behavior of modulo multiplication without the clear algebraic structure. Another object and advantage of the form of multiplication is enabling folding the result of the form

of multiplication with itself or another companion execution. Another object and advantage is that the machine can be generalized to more than two arguments.

In the mentioned preferred embodiment of the machine of the present invention, the form of multiplication in the key inserter comprises: common multiplication of arguments to yield a first product, common multiplication of other arguments to yield a second product, designating a upper and lower half of the first product, designating an upper and lower half of the second product, combining the upper half of the first product with the lower half of the second product using exclusive-or to form a first final product. Combining the upper half of the second product with the lower half of the first product using exclusive-or to form a second final product. An object and advantage of the form of multiplication is that the resultant apparatus for folding solves the long-felt need for a 128-bit block method. Another advantage of the form of multiplication is that the machine can be generalized to more than two arguments.

In the mentioned preferred embodiment of the machine of the present invention, the form of multiplication in the key inserter comprises: circuits to perform multiplication on a plurality of arguments to form a first product. Logic circuits perform exclusive-or on the plurality of arguments to form a second product. Logic circuits to perform addition between the first product and the second product to form a gorilla product. An advantage of the form of multiplication is that the result is a pseudo-random expansion of one of the arguments. An object of the form of multiplication is that it enables folding the result of the form of multiplication with itself or another companion execution. Another advantage of the form of multiplication is that the machine can be generalized to more than two arguments. A preferred embodiment of the folding machine of the present invention wherein the gorilla product is provided to a machine comprising: a counter which counts the plurality of arguments, calling it *n*. A repeater provides a new set of arguments and calculates *n* gorilla products. A splitter which divides each gorilla product into *n* pieces, each with index *i* from 1..n. A combiner which combines using exclusive-or *n* pieces such that the combine will take exactly one piece from each gorilla product, and exactly one piece of any gorilla product with the index *i* for all I. The combiner yields a plurality of *n* folded products. A preferred embodiment of the machine of the present invention, wherein the form of multiplication in the key inserter comprises: (a*b) | (a exclusive-or b), whereby the result is a pseudo-random expansion of one of the arguments.

Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt comprising: employing an operation on two inputs yielding a double-size result, folding half of result into a companion execution. An advantage and object is that a shorter input length keyed hash function can be built. Another advantage is that pass phrases can be processed without excessive padding.

Reference is made to figure 4. A preferred embodiment of the method of the present invention for constructing a key schedule for an encryption algorithm, the steps of the method being performed by a data processor, the method comprising the steps of: determining a first set of at least one subkey for the encryption algorithm; encrypting a master key according to the encryption algorithm by using first set of at least one subkey to product a cipher text, repeating the encryption of the master key for at least a first number of rounds required to achieve dependence of every bit of cipher text on each bit of master key; continuing the encryption of the master key for an integral number of rounds, integral number being at least one, extracting subkeys from the output of the round, further continuing the encryption of the master key and extraction of subkeys until a second set of subkeys has been generated. An advantage and object is that the key schedule solves the need for an expandable, generalizable, fast, user defined speed, well-mixed key schedule.

A preferred embodiment of the method of the present invention wherein the first set of at least one subkey is derived from DES s-box entries.

A preferred embodiment of the method of the present invention wherein the second set of at least one subkey is derived from the output of the round function in the encryption algorithm.

A preferred embodiment of the method of the present invention further comprising the steps of: encrypting the cipher text with the second set of at least one subkey according to the encryption algorithm to produce further encrypted cipher text, with the object and advantage of creating a third set of subkeys for use in encryption of actual plain text.

Reference is made to figure 5. Another preferred embodiment of the machine of the present invention for encrypting comprising: circuits which employ at least a 128-bit key and block size. An object and advantage is that the machine is suitable as a hash function. An additional advantage is employing the current invention instead of a human needing to provide and debug a distinct, less well

understood specialized hash. An unexpected result is that every bit of key and every bit of plain text cause every single bit of the resultant cipher text to become unpredictable.

Another preferred embodiment of the machine of the present invention for encrypting further comprising the circuits providing the large key size are implemented by using the circuits providing a large block size. An advantage is that the machine key schedule can be accomplished in zero additional time. An object is that the machine mixes rapidly over the entire block size. Another advantage is the generality of the key schedule which provides a rapid key schedule design ready for new ciphers.

Another preferred embodiment of the machine of the present invention for encrypting further comprising an optimal sorting network. An advantage of employing an optimal sorting network is to ensure complete mixing within each round. An object of employing the generalized construction of optimal sorting methods allows the machine to be extended to arbitrary sizes. An advantage of accomplishing extension of block size to arbitrary sizes allows larger proportions of the output to be disclosed together, yet reversal of the whole process remains difficult.

Reference is made to figure 6. Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt employing a key schedule comprising: feeding the full set of 64 key bits per block into a rearranged PC2 from DES. An object of feeding the full 64 bits per block into a rearranged PC2 from DES is that all of the key bits provided by the user are employed. An advantage of employing all of the key bits is that exhaustive search on such a modified method would require guessing the full 64 bits. For a number of years, attempts have failed to generate an accepted key schedule that solves the long-felt need for using the all the bits in the masterkey.

Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt employing a key schedule further comprising: entries of PC2 with values above 28 have four added to them. An object of adding four to values above 28 is that a schedule will be balanced left and right halves. An advantage of a selected key table (see specifically figure 10, table I) is that round subkey bits depend equally on any given master key bit.

Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt employing a key schedule further comprising: the key schedule rotation is carried out 64 bits at a time rather than in two groups of 32 each, with an advantage of eliminating the distinction between two halves present in the prior art. An object is an eavesdropper would find it more difficult to isolate parts of a key.

Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt employing a key schedule further comprising: the subkey is made dependent on the serial number of the parallel execution, with an advantage that even if masterkey repeats exactly that subkeys will not. An object of causing output from a system with a repeated master key and repeated data to be distinct, causes the typical demonstration of a product built according to the method would be more pleasant to humans.

Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt employing a key schedule further comprising: the subkey used is derived from finding a multiplicative inverse over a field, with an advantage that the key insertion operation becomes thereby modulo division. An object of modulo division is that such a key insertion operation is no longer argument order insensitive. An advantage of the order sensitivity is that interchanging plain text and master key give different results, even for a key insertion operation.

Another preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable data processor to encrypt employing a key schedule further comprising: the zero sub key is replaced by a round dependent mask value. An advantage of using a round dependent mask is that weak keys are replaced with arbitrary and better values. An object of employing a round dependent mask value is that the typically demonstrated zero master key provides a decent mixing function.

Reference is made to figure 7. Another preferred and optional embodiment of the method of the present invention described in figure 1, wherein the form of multiplication features the steps of: multiplying a plurality of bits from the plain text-derived input and a plurality of bits from the plurality of round-dependent sub keys to form a common multiplication product; performing an exclusive-or function on a plurality of bits from the plain text-derived input and a plurality of bits from the

plurality of round-dependent sub keys to form a balanced product. The step of combining the plain text-derived input with a plurality of round-dependent sub keys further comprises the steps of: performing an addition function on the common multiplication product and the balanced product to form a pseudo-random product. The step of combining the plain text-derived input with a plurality of round-dependent sub keys further comprises the steps of performing a thorough folding operation on two pseudo-random products as follows: fold the upper half of the first pseudo random product into the lower half of the second pseudo random product to form first result. fold the lower half of the first pseudo random product into the upper half of the second pseudo random product to form second result. Concatenate first result to second result to form a folded product. An advantage of these or equivalent steps is that all the bits of each of the products depends heavily on both plain text-derived inputs and both round-dependent sub keys. The step of combining the plain text-derived input with a plurality of round-dependent sub keys further comprises the steps of performing a blending operation on two folded products as follows: concatenate lower half of the first folded product with upper half of second folded product to form a blended product, optionally and preferably, fold operation is exclusive-or. An object is an input to a plurality of distinct s-boxes depends on four plain text derived inputs and four corresponding round-dependent sub keys. These advantages and objects have many alternative descriptions, any description with similar results is sufficient. The descriptions provided herein are strictly exemplary.

Reference is made to figure 8. An alternative embodiment of the machine of the present invention employs a extended P Permutation machine comprising a local scrambling operation and a permutation distributing bits from output of a given local scrambler to input of other local scramblers.

An extended P permutation is defined as a permutation on groups of s-boxes wherein the orbital property is preserved between (and within) the groups of s-boxes. Where the orbital property is not possible, because the number of outputs is limited, an extended P permutation will distribute the output bits evenly, balancing value of public bits against private bits to break symmetry. Public bits are those repeated by the E expansion. A machine for data scrambling comprising a local scrambling operation and a permutation distributing bits from output of a given local scrambler to input of other local scramblers, comprising: a local scrambler P which distributes four outputs among eight possible boxes, and a global scrambler PP which distributes a plurality of outputs among groups of possible

s-boxes to effect an extended P permutation. Optionally and alternatively, a known permutation is used within each scrambler, further comprising: wires which interconnect the output of a given scrambler with inputs of other scramblers. Optionally and alternatively, known permutation is the prior art P permutation from DES.

Reference is made to figure 9. A preferred embodiment of the method of the present invention for implementing substitution boxes in logic gates on a 32-bit microprocessor. An advantage of employing a 32-bit processor is that the method is applicable to Intel compatible microprocessors.

Another preferred embodiment of the method of the present invention wherein the plurality of s-boxes are applied in bit-slice form using logic gates. An object and advantage is that a physical apparatus can be easily built and speed gains achieved.

A preferred embodiment of the machine of the present invention wherein the encrypting is implemented by bit-slicing circuits. An advantage is thereby providing a design for a physical apparatus of logic gates. An object is the machine can be implemented with fivefold speed gains.

Reference is made to figure 10. A preferred embodiment of the method of the present invention for operating a general purpose data processor of known type to enable the data processor to encrypt comprising: employing masks in which the mask used depends on information available within the round function selected from the group consisting of round number and data being encrypted, with an advantage that a repeated plain text-derived-input sub key pairs will still permit the round function to generate distinct output. An object is to correctly treat a master key with repeated segments useful to verify the functionality of the method.

Another preferred embodiment of the method of the present invention, optionally and preferably further comprising the step of performing a combining operation on the plurality of input blocks with a mask determined according to a criteria selected from the group of a number of a round being performed and the input block number.

A preferred embodiment of the machine of the present invention for encrypting plain text-derived-input comprising: a memory providing the s-boxes of DES as numbers a logic circuit which combines the numbers on a bit-by-bit basis with limited carry into the stream of the plain text-derived-input. Exclusive-or is a group-like operation relative to Hamming weights. Given one input with a given Hamming weight, it is always possible to find a second input such that the output Hamming weight

will be that desired. If one of the arguments has a balanced Hamming weight, approximately equal number of zeros and ones, and the other argument has an unbalanced Hamming weight, mostly zeros or mostly ones, the result will usually be more balanced than the second argument. Thus, an advantage is that Hamming weights plain text derived input which tend toward unbalanced hamming weights will have that tendency corrected. An object of corrected unbalanced input is to allow providing plain text with redundancy.

Reference is made to figure 11. A preferred embodiment of the machine of the present invention for an operation selected from the group of hashing machine and encryptor wherein a plain text and a plurality of sub keys are employed as new sub key generators to generate new sub keys, whereby the new sub keys are employed to process future plain texts.

One embodiment of the method of the present invention seeks to provide improved methods for DES encryption. It being understood that an interative block cipher could be used in place of DES.

It being further understood that any cryptographic primitive could be used to replace DES in the description and claims. Moreover, whereever addition or multiplication are used in the claims, it is illustrative being that either could be replaced by at least one operation selected from the group consisting of a form of multiplication, blending, folding and combining.

Reference is made to figures 20-26. There is thus provided, in accordance with a preferred embodiment of the present invention, a method for protecting confidentiality of information written on a notebook computer the method comprising: protecting a plurality of files on an automatic file-by-file basis, wherein protection of each individual file includes the following steps: using a symmetric cryptosystem to encrypt the individual file, thereby to generate an encrypted individual file; and storing the encrypted individual file on the notebook computer.

Also provided, in accordance with another preferred embodiment of the present invention is a method for protecting confidentiality of information written on a hard disk, the method comprising: using a first symmetric cryptosystem to encrypt a file having a selectably known file key; and encrypting the selectably known file key using a second symmetric cryptosystem and a selectably known master key derived from a selectably known pass phrase using a cryptographically strong hash function.

Further in accordance with another preferred embodiment of the present invention is a method comprising the following steps: decrypting the selectably known file key using the second symmetric cryptosystem and the selectably known masterkey; and decrypting the file using the selectably known file key and the first symmetric cryptosystem. Further in accordance with another preferred embodiment of the present invention is a method in which the cryptographically strong hash function comprises a MAC (message authentication code).

Reference is made to figures 27-36. There is thus provided, in accordance with a preferred embodiment of the present invention, a DES encryption method including performing N DES rounds, including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to the n'th round wherein addition is substituted for exclusive-or in performing the n'th DES round, wherein a sub key is defined for each of the N rounds and wherein at least some of the N sub keys are dependent.

Further, in accordance with a preferred embodiment of the present invention, all of the N sub keys are derived from a standard key schedule. Still further in accordance with a preferred embodiment of the present invention the plain text derived input to the n'th round (n>1) comprises an output of a round previous to the n'th round. Additionally, in accordance with a preferred embodiment of the present invention, the plain text derived input to the first round comprises at least a portion of the plain text.

Also provided, in accordance with another preferred embodiment of the present invention, is a DES encryption method including performing N>16 rounds, including for at least one 1 <= n <= N, performing an n'th DES round on a sub key and a plain text derived input to the n'th round wherein addition is substituted for exclusive-or in performing the n'th DES round.

Also provided, in accordance with another preferred embodiment of the present invention, is a DES encryption system including an addition-based DES encryptor operative to perform N DES rounds including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to the n'th round wherein addition rather than exclusive-or is used to perform the n'th DES round, wherein a sub key is defined for each of the N rounds and wherein at least some of the N sub keys are dependent. Further in accordance with a preferred embodiment of the present

invention, the step of performing N DES rounds comprises performing a bit-slice implementation of DES.

Also provided, in accordance with another preferred embodiment of the present invention, is a DES encryption method comprising: performing N DES rounds, including, for at least one 1<= n <= N, performing an n'th DES round on a sub key and a plain-text derived input to the n'th round, wherein the step of performing N DES rounds comprises using a personal computer to perform a bit-slice implementation of DES. Further in accordance with a preferred embodiment of the present invention, the personal computer has at least one register which is 32-bits long.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: performing N DES rounds, including, for at least one 1 <= n <= N, performing an n'th DES round on a sub key and a plain text derived input to the n'th round, wherein the step of performing N DES rounds comprises using a computer having registers whose size is less than 64 bits to perform bit-slice implementation of DES.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: computing a sub key for each of N DES rounds, at least some of the N sub keys being dependent, by combining a plurality of key to sub key operations into a single key to sub key operation on a DES key, thereby to provide a sub key; and performing N DES rounds. Further in accordance with a preferred embodiment of the present invention, for at least one 1<=n<=N, the step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key, is performed before the (n+1)th round is performed. Further in accordance with a preferred embodiment of the present invention, for at least one 1<=n<=N, the step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key is performed before the n'th round is performed. Further in accordance with a preferred embodiment of the present invention, for at least one 1<=n<=N, the step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key is performed before the n'th sub key is used. Further in accordance with a preferred embodiment of the present invention, for at least one 1<=n<=N, the step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key is performed before completing the use of the n'th sub key.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: using first and second permutations and a mapping to perform each of N DES rounds, wherein the first permutation includes a left half of L* and a right half R* and wherein L* comprises a composition of an inverse P permutation and a left half, L, of an initial permutation, and wherein R*, comprises a composition of the inverse P permutation and a right half, R, of the initial permutation, wherein the second permutation includes a left half of L** and a right half R** and wherein L** comprises a composition of the P permutation and a left half of the final permutation., and R** comprises a composition of the P permutation and a right half of the final permutation, and, wherein the mapping comprises a composition of the P permutation with an E expansion.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: performing N DES rounds, including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to the n'th round wherein addition is substituted for exclusive-or in performing the n'th DES round, wherein the step of performing N DES rounds comprises performing a bit-slice implementation of DES.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: performing N DES rounds, including, for at least one 1<=n<=N, generating an n'th k-bit s-box input by performing an n'th DES round on a k-bit sub key and a k-bit plain text derived input to the n'th round wherein multiplication in which any carry beyond k bits is discarded, is substituted for exclusive-or in performing the n'th DES round. Further in accordance with a preferred embodiment of the present invention, all of the N sub keys are derived from a standard key schedule. Further in accordance with a preferred embodiment of the present invention, the plain-text derived input to the n'th round (n>1) comprises an output of a round previous to the n'th round. Further in accordance with a preferred embodiment of the present invention, the plain text derived input to the first round comprises at least a portion of the plain text. Further in accordance with a preferred embodiment of the present invention, N>16.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor operative to perform N>16 DES rounds, including, for at least one 1<= n <=N, performing an n'th DES round on a sub key and a plain text

derived input to the n'th round wherein addition is substituted for exclusive-or in performing the n'th DES round. Further in accordance with a preferred embodiment of the present invention, the step of performing an n'th DES round comprises performing a bit-slice DES round.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: performing N DES rounds, including, for at least one 1<=n <=N, performing an n'th DES round on a sub key and a plain text derived input to the n'th round wherein addition is substituted for exclusive-or in performing the n'th DES round, wherein the step of performing N DES rounds comprises performing a bit-slice implementation of DES. Further in accordance with a preferred embodiment of the present invention, wherein a sub key is defined for each of the N rounds and wherein at least some of the N sub keys are dependent.

Also provided, in accordance with another preferred embodiment of the present invention is a WDES encryption method comprising: performing a plurality of rounds of WDES encryption each round using a round function F; wherein, for the round function F of a t least one round, addition, with final carry neglected is substituted for exclusive or.

Also provided, in accordance with another preferred embodiment of the present invention is a WDES encryption method comprising: performing a plurality of rounds of WDES encryption each round using a round function F; wherein, for the round function F of at least one round, a form of multiplication is substituted for exclusive-or.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption method comprising: performing N DES rounds, including for at least one 1<=n <=N, generating an n'th k-bit s-box input by performing an n'th DES round on a k-bit sub key and a k-bit plain text derived input to the n'th round wherein multiplication, performed over a ring, is substituted for exclusive-or in performing the n'th DES round. Further in accordance with a preferred embodiment of the present invention, herein the multiplication over a ring comprises multiplication over a finite field. Further in accordance with a preferred embodiment of the present invention, wherein the ring has a modulus and the modulus is a product of less than 5 primes. Further in accordance with a preferred embodiment of the present invention, wherein the ring has a modulus and the modulus is a product of less than 4 primes. Further in accordance with a preferred embodiment of the present invention, the ring has a modulus and the modulus is a product of 2 primes. Further in

accordance with a preferred embodiment of the present invention, the ring has a modulus and the modulus is prime. Further in accordance with a preferred embodiment of the present invention, the ring has a modulus and the modulus comprises a product of a plurality of primes at least one of which slightly exceeds an exponent of 256. Further in accordance with a preferred embodiment of the present invention, the ring has a modulus and the modulus comprises a product of a plurality of primes at least one of which slightly exceeds an exponent of 65536 such as 65536 or $2^{32}$ or $2^{48}$ or $2^{64}$. Further in accordance with a preferred embodiment of the present invention, the ring has a modulus and the modulus comprises a product of a plurality of primes at least one of which slightly less than an exponent of 256. Further in accordance with a preferred embodiment of the present invention, wherein the ring has a modulus and the modulus comprises a product of a plurality of primes at least one of which slightly less than an exponent of 65536 such as 65536 or $2^{32}$ or $2^{48}$ or $2^{64}$.

Also provided, in accordance with another preferred embodiment of the present invention is a WDES encryption method comprising: performing a plurality of rounds of WDES encryption, each using a round function F; wherein, for the round function F of at least one round, multiplication over a ring is substituted for exclusive or. Further in accordance with a preferred embodiment of the present invention, the step of performing an n'th DES round comprises performing a bit-slice DES round.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor for performing N>16 DES rounds, including, for at least one 1<= n<=N, an addition-based DES engine operative to perform an n'th DES round on a sub key and a plain text derived input to the n'th round wherein addition rather than exclusive or is sued to perform the n'th DES round.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor for performing N DES rounds, including, for at least one 1<=n<=N, a DES engine operative to perform an n'th DES round on a sub key and a plain text derived input to the n'th round; and a computer having registers whose size is less than 64 bits, wherein the DES encryptor is configured to perform the N DES round including performing a bit-slice implementation of DES while running on the computer.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a sub key computation engine operative to compute a sub key for each of N DES rounds, at least some of the N sub keys being dependent, the sub key computation engine including a single key-to-sub key operator performing a combination of a plurality of key-to-sub key operations as a single key-to-sub key operation and performing the single key-to-sub key operation on a DES key, thereby to provide a sub key; and a DES engine operative to perform N DES rounds using the N sub keys.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor using first and second permutations and a mapping to perform each of N DES rounds, the DES encryptor comprising: a first permutation provider providing the first permutation which includes a left half L* and a right half R* and wherein L* comprises a composition of an inverse P permutation and a left half L of an initial permutation, and wherein R* comprises a composition of an inverse P permutation and a right half R of an initial permutation, a second permutation provider providing the first permutation which includes a left half L** and a right half R** wherein L** comprises a composition of the P permutation and a left half L of a final permutation, and wherein R** comprises a composition of the P permutation and a right half R of a final permutation, and a mapping provider providing the mapping which comprises a composition of the P permutation and the E expansion.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor operative to perform N DES rounds, including an addition-based DES engine performing, for at least one 1<=n<=N, an n'th DES round on a sub key and a plain-text derived input to the n'th round wherein addition rather than exclusive or is used in performing the n'th DES round, wherein the N DES rounds are performed by performing a bit-slice implementation of DES.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor operative to perform N DES rounds, including an s-box input provider operative to provide for at least one 1<=n<=N an n'th k-bit s-box input by performing an n'th DES round on an k-bit sub key and a k-bit plain text derived input to the n'th

round wherein multiplication with any carry beyond k bits is discarded, is used, rather than using exclusive or in performing the n'th DES round.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor operative to perform N DES rounds, including an addition-based DES engine operative, for at least one 1<=n<=N, to perform an n'th DES round on a sub key and a plain text-derived-input to the n'th round wherein addition rather than exclusive or is used in performing a bit-slice implementation of DES.

Also provided, in accordance with another preferred embodiment of the present invention is a WDES encryption system comprising: a WDES encryptor operative to perform a plurality of rounds of WDES encryption, each round using a round function F, the WDES encryptor including an addition-based WDES engine operative for the round function F the WDES encryptor of at least one round to perform addition with final carry neglected rather than performing exclusive or.

Also provided, in accordance with another preferred embodiment of the present invention is a WDES encryption system comprising: a WDES encryptor operative to perform a plurality of rounds of WDES encryption, each round using a round function F, the WDES encryptor including a common multiplication-based WDES engine operative for the round function F of at least one round to perform common multiplication with final carry neglected rather than performing exclusive-or.

Also provided, in accordance with another preferred embodiment of the present invention is a DES encryption system comprising: a DES encryptor operative to perform N DES rounds, the DES encryptor including, for at least one 1<=n<=N, an s-box input provider operative to provide an n'th k-bit s-box input by performing an n'th DES round on a k-bit sub key and a k-bit plain text derived input to the n'th round wherein the n'th DES round includes performing multiplication over a ring rather than performing exclusive-or.

It is appreciated that the number of bits used to store any of the various quantities shown and described herein need not necessarily be exactly as described herein. Typically, the multiplicative ratio between the various number of bits used to store various quantities within a particular method, remains constant even if the quantities themselves are varied.

## DESCRIPTION OF AN EMBODIMENT OF INVENTION: MULTIDES

A personal computer refers to a wide variety of computers whose architecture is similar to the IBM PC architecture. The term "personal computer" is not intended to include minicomputers such as a DEC-Alpha.

The term "bit-slice DES" refers to the encryption methods shown and suggested in Biham, E., "A fast new DES implementation in software," Proceedings of Fast Software Encryption Workshop, Springer-Verlag, January 1997, and to known equivalents of the methods shown and suggested by E. Biham.

*BASIC KEY INSERTION OPERATION*

Figure 14, explained in more detail later, is an exemplary illustration of a preferred embodiment which explains how to make and use an internal round function portion of the method of the present invention. Expansion, s-boxes and P-permutation are as appearing in prior art DES. One object of the method of the present invention is to overcome weaknesses in prior art DES which caused it to succumb to differential cryptanalysis.

Since the introduction of the prior art Data Encryption Standard, there has been interest in its strength and design criteria. With the discovery of prior art differential cryptanalysis the optimality of certain aspects of the design became apparent. Central weaknesses of the prior art include bit-wise independent operations and use of involution for key insertion. In proposing any cipher, the burden lies with the authors to show that it withstands these now classical attacks well.

Upon a close reading of the results of Biham-Shamir [BiSh93] and extensive analysis, it became clear that the bit-wise involution for combining the subkey inside the F function was not a feature that strengthened the cipher.

The inventor discovered that these two alternative attributes (a) bit-wise and (b) involution were responsible for the success and simplicity of differential cryptanalysis. (Eurocrypt '98: Properties of DES that facilitate Differential Cryptanalysis, Stiebel, J.)

The prior art is vulnerable because of use of bit-wise involution. The bit-wise aspect allowed for commutativity between the permutations and the key-insertion operation. Likewise, differential cryptanalysis is able to effectively "ignore" the E expansion and P permutation.

Differential cryptanalysis deals with the question of how to overcome the S substitution boxes using input exclusive-or, probability and output exclusive-or.

The use of the involution enables canceling the effect of the round-key.

Preferably, replacement of a bit-wise operation such as exclusive-or with a group operation such as a form of multiplication, optionally over a ring or field, enhances the cryptographic strength of the cipher.

The existing F function used in the round is documented in the prior art DES patent. The operations used in the prior art F function are E expansion, key insertion with exclusive-or, s-box calculation, followed by P permutation. (See Tables II, III, and IV below.)

**KEY INSERTION OPERATION**

Reference is made to figures 1, and 2. Figure 2 is an exemplary illustration of a preferred embodiment which explains how to make and use the key insertion portion of the method of the present invention. Preferably, a form of multiplication chosen will be the common product of the two arguments plus exclusive or of the arguments. Optionally, modulo multiplication employs over a Fermat field. Preferably and optionally, an embodiment of the method of the present invention defines the form of multiplication to be common multiplication with upper and lower halves folded together. Preferably and optionally, an embodiment of the method of the present invention defines the form of multiplication to be common multiplication with an upper folded into a lower half of a companion execution of the method. Preferably and optionally, the method of the current invention is employed in a bit-slice implementation of the s-boxes.

Exclusive-or is a bit-wise involution. Exclusive-or is a simpler operation which can model addition. Not only is exclusive-or commutative (unlike subtraction), but it is also self-canceling.

Although exclusive-or causes every bit of the output to depend on bits of each argument to the exclusive-or, the effect is extremely localized. It depends on exactly ONE bit of each of the arguments.

This characteristic of bit-wise operation allows exclusive-or, as well as by extension the input exclusive-or used for differential cryptanalysis to commute with the P Permutation and E Expansion found in DES.

Thus, P Permutation is typically combined with the E Expansion of the following round.

Optionally, the P Permutation is combined with the s-boxes. More interestingly, the E Expansion is combined with the s-boxes of the current round (not the previous round).

In the method of the present invention, the optimizations of combining the E Expansion with the previous or current round are no longer equivalent. Hence, in the preferred embodiment of the method of the present invention MultiDES, perform the E Expansion after the multiplication and folding. In an alternative embodiment of the method of the present invention as exemplified by MultiDES based systems with bit-slice implementation, perform the E Expansion prior to the multiplication and folding.

For example, an operation that approximates a group is preferred. This property means given an output, for any given input specified for argument A, there exists an argument B such that A <group-operation> B is the output.

Alternatively, the operation should not be an involution (self-canceling). The weakness of such a property is well known. This holds even if the involution is exclusive-or with a completely unknown random string.

**CRYPTOBOX, PLURALITY-SIZE RESULT**

Reference is made to figures 1 and 3. A cryptographic primative refers to a wide variety of operations whose goals or methods are similar to hashing, encrypting, decrypting, digital signatures, key generation, substitution, permutation or identification, hereinafter referred to as an encryption method. A cryptographic processor is a machine which performs a cryptographic primative, hereinafter referred to as a "cryptobox." Although, for clarity, the method of the present invention is described alternately as a hash, as an encryption function, and as a key-generation mechanism, it is understood by one skilled in the art that such choice of description in the case of the present invention is strictly illustrative and in no means meant to be limiting to one form of cryptographic primative or another.

A plurality of inputs designates at least one input. A plurality size result is a result of size equavalent to concatenation of the plurality of inputs.. A single size portion is size of a single input.

For example, let the plurality be two. Thus, a double-size input yields a double-size result. One half of the result is a single-size portion. Alternatively, let the plurality be three. Thus, a triple-size input yields a triple-size result. One third of the result is a single-size portion. Let the plurality, hereinafter, be any natural number.

A companion execution refers either to a parallel execution of an embodiment of the invention or to its own execution. In cases wherein there is only one execution it refers to that execution.

*STRENGTHENING KEYS and PLAIN TEXTS: MASKS*

Reference is made to figures 1 and 10. In the preferred embodiment of the method of the present invention, to prevent identical plain text inputs together with identical subkeys from yielding identical round-end cipher text, introduce an exclusive-or mask with a constant value which is evenly balanced zeros and ones. The exclusive-or mask is typically depending on up to two elements selected from the set of the round number and block number. The round number is the cardinal number of the round. The block number is the cardinal number of the basic half block unit size such as 32 bits. Typically, this is done prior to the key insertion operation. Alternatively, the exclusive-or mask is employed adjacent to s-box application.

In the preferred embodiment of the method of the present invention, keys or plain texts whose Hamming weight tends towards maximum or minimum possible for given key or block size may have incomplete mixing properties when using a common or modular multiplication operation. Preferably, to ensure more thorough mixing and a plain text or key independent preservation of entropy entering the round, the traditional exclusive-or of the subkey and the plain text derived input is added to the product of the subkey and the plain text derived input.

Reference is made to figure 1. The method of the present invention whose different embodiments are *MultiDES based systems* and *MultiDES based systems with bit-slice implementation* share steps:

0. Optionally, number the 32-bit input blocks 0..n; split each block into upper and lower halves. (This step is strictly for notation.) This is box 110 in figure 1.

Preferably, output of the optional step (continue to call it "plain text derived input"), or of the plain text derived input directly, is combined with round-number dependent subkeys. Preferably, each piece of the plain text derived input is used exactly once.

1. Optionally, exclusive-or plain text derived input with round and input block number dependent mask. In the preferred embodiment of the method of the current invention, derive the mask from the s-boxes as shown in (figure 10, described below). This step is optional. This is box 120 in figure 1.

2. Preferably, employ a form of multiplication to combine plain text derived input (output from step 0 or 1) with round dependent subkeys. The form of multiplication used in the preferred

embodiment includes common multiplication of the two arguments plus exclusive-or of the two arguments. This is box 130 in figure 1.

3. Preferably, fold the result of two multiplications together. The form of folding used in the preferred embodiment is exclusive-or upper half of one multiplication with the lower half of the other. Concatenate the results to form a full-size number. This is box 140 in figure 1.

4. Preferably, blend the result of the previous folding to effect folding of four distinct multiplications together. The form of blending used in the preferred embodiment is the concatenation of the lower half of the first argument with the upper half of the second argument. This is box 150 in figure 1.

Preferably, after blending two products, for example, blend($a,b$), then blend the same two products again, for example, blend($b,a$).

5. The preferred embodiment of the present invention employs the E expansion mapping just immediately before the s-boxes. A preferable bit-slice embodiment of the present invention employs the E expansion mapping just immediately prior to the multiplication step. This is step 160 in figure 1. In the preferred embodiment of the present invention, combine the P permutation with either the E expansion or the s-boxes. This is step 170 in figure 1.

6. Preferably, the s-boxes are then performed either normally or in bit-slice form using logic gates

7. Preferably, apply the P-based permutation. This is step 170 in figure 1.

Although the cipher preserves the Feistel structure, the principles herein apply also to non-Feistel ciphers. For example, an exemplary embodiment of the method of the present invention in which the round function influences and receives influence from at least half of the bits of the block size. These ideas are relevant, for example, to IDEA and to JADE.

Figure 3 is an exemplary illustration of a preferred embodiment which explains how to make and use the operation on two inputs yielding a double-sized result portion of the method of the current invention. Preferably, half of the double-sized result is folded into a companion execution of the method.

Optionally, an operation is employed on $n$ inputs yielding a $n$ sized result folding $n$-$1$ pieces of the result into a companion execution. Optionally, each input in the folding is determined to come from a

different relative position with the n-sized result. Optionally, an operation is employed at least once on two inputs to yield a double sized result in order to mix two distinct arguments.

**Key Schedule**

Reference is made to figures 4-6. In this section, the two embodiments of the method of the present invention are detailed as regards the key schedule. The first key schedule embodiment differs from the prior art key schedule by constructing the round subkeys with all of the bits of the master key. The preferred key schedule embodiment of the present invention uses the block cipher itself to generate the subkeys such that each bit of any subkey depends on every bit of the master key.

**SELECTED KEY TABLE**

After a careful examination of the currently available implementations of the prior art DES key schedule, disturbing properties were noted. Reference is made specifically to figure 6.

The entire prior art schedule amounts to selection of two groups of 28 bits from the master key of 56 bits plus 8 parity bits. Two permutations are applied. Each subkey bit is exactly one bit of the master key. Each half of every subkey is derived from a distinct half of the master key. Only 56 bits of the available 64 bits are used. The key size is different than the block size, resulting in cryptographic modes which have dangerous short cycle properties. Because the key schedule permutes individual bits, it is particularly slow in software.

As a first approach, the parity bits are eliminated. Thus, all 64 bits are available for selection in the 48-bit round subkeys. Thus, rotations in the key schedule operate on a full 32 bits each rather than 28 bits a piece in the prior art.

Next, the two permutations "Permuted Choice 1" (hereinafter PC1) and "Permuted Choice 2" (hereinafter PC2) may be composed using standard combinatorial methods wherein application of the composition is equivalent to the application of PC1 followed by PC2. (See figure 10, table I; figure 6, box 610)

Then, note that the resultant composition of PC1 and PC2 selects from only the first 56 bits of the master key (discounting rotations in the key schedule). Including rotations, the lower 32 bits have a higher probability to be included in the round subkeys.

Thus, add four to each entry in the resultant table wherein the entry in prior art would refer to bits 29-56. This makes the number of bits selected in each half of the subkeys from the master key equal in number for the two halves. (See figure 6, box 620.)

The order of the rows are preferably rearranged in the resultant table. The purpose of the rearrangement is to cause every second row to refer to bits above the half-way mark while the other half of the rows refer to bits below the half way mark.

Such a table can be referred to as the "key selection permutation table (see figure 10, table 1). Such a key schedule can be referred to as "improved." (See figure 6.)

A typical key schedule for DES, MultiDES based systems, according to one embodiment of the method of the present invention, would use the selected key table to generate subkey bits from the master key. The improved key schedule thereby employs a full 64 bits, uses only a single permutation, cancels separation wherein upper halves of master key corresponded to upper half of subkey bits.

Preferably, the method further includes the step of feeding the full 64 key bits per block into a rearranged PC2 from prior art DES, whereby all key bits provided by user are employed. Optionally, entries of PC2 with values above 28 have a value four added to them. These steps ensure that the key schedule will be balanced at the left and right halves. Optionally, the key schedule rotation is carried out a block at a time rather than in two half block groupings. Optionally, subkey is made dependent on the serial number of parallel execution. Thus, even if master key contains exact repeating sequences, subkeys will not necessarily repeat. Optionally, subkey employed is derived by finding a multiplicative inverse over a field. Optionally, zero subkey is replaced by a round dependent mask value.

**BOOTSTRAP KEY SCHEDULE**

The entire schedule amounts to selection of two groups of 28 bits from the master key of 56 bits plus 8 parity bits. Two permutations are applied. Each subkey bit is exactly one bit of the master key. Each half of every subkey is derived from a distinct half of the master key. Only 56 bits of the available 64 bits are used. Key size is different than the block size, resulting in cryptographic modes which have dangerous short cycle properties. Because key schedule permutes individual bits, it is particularly slow in software.

Reference is made specifically to figure 5. Preferably, a large key size is implemented using a large block size. Optionally, a larger key size is accommodated by employing cipher-block-chaining while generating the keys. Optionally, a larger key size is accommodated by employing an embodiment of the invention of that block size to generate subkeys. Optionally, the delay between generating subkeys can be made arbitrarily long with the object and advantage to increase necessary time for exhaustive key search for a given key size. Optionally, an optimal sorting circuit design is used to determine how to perform the pairings for the foldings within the round. Large key size or large block size is understood to be at least 128 bits long.

Reference is made specifically to figure 4. After examination of the improved key schedule, the first key schedule embodiment of the present invention has the following properties: the rotation amount between rounds is unchanged; in this embodiment, the method is restricted to work on units of 64 bits at a time; and the schedule yields exactly 48 bits for each subkey and most importantly is still a permutation on master key bits.

A desired property of any key schedule would be to cause a change in a single bit in master key to cause about half the bits of the result subkeys produced to be flipped. Additionally, each subkey bit should be computationally independent from any given bit in master key.

To accomplish this second approach, assume a strong block cipher with following properties. After four rounds of encryption, every bit of the output depends on each bit of input. Preferably, the block size is at least desired master key size. Assume that encryption under any given key yields cipher text which is computationally indistinguishable from a random permutation.

Thus, recommended key schedule is composed as follows. (Reference is made to figure 4.)

1. Set encryption algorithm to use a set of subkeys which are master key independent. This is step 410 in figure 4. The preferred embodiment of the key-schedule method of the present invention employs, optionally and preferably, the subkeys independent of the master key and the subkeys derived from values in the s-boxes. Usage of such strings from the s-boxes provides easily available set of numbers that are known to be a permutation. Moreover, the explanation of the choice imparts a higher level of confidence in the method of the current invention to those of ordinary skill in the art.

2. Encrypt desired master key at least number of rounds to achieve dependence of every bit of cipher text on each bit of master key. Typically, this is four rounds. This is step 420 in figure 4.

3. Encrypt further an integral number of rounds, typically 1, 4, 8 or 16. Use output of s-boxes as desired subkeys. Repeat previous step until sufficient subkey material is pseudo-randomly generated for all the rounds, typically 16 rounds are employed. Although, substantially any number could be used. Major benefits of Feistel structure completeness are realized already with these values. This is step 430-440 in figure 4. Sample and store the key material after each employment of an integral number of rounds as step 430 in figure 4. Without executing the cipher again, it would be difficult for a key-search attack to determine whether the guessed key was correct. Thus, the method of the present invention provides additional security against exhaustive search attacks which the prior art DES is vulnerable.

4. Optionally and preferably, repeat step 2 using subkeys generated in step 3. This is step 450 in figure 4. Optionally, at least once set the encryption keys to be the subkeys generated and encrypt the cipher text generated, to yield a new set of encryption keys.

An object and advantage of the method of the present invention is use of avalanche effect whereby after four rounds, preferably and optionally, any specific input bit will affect any specific output bit. The bootstrap key schedule provides the feature of the method of the present invention that encryption is rapid unlike the prior art DES patent wherein each bit was handled individually.

An object and advantage of the key schedule is effective operation even using just a single key bit, since the output of the subkeys will be changed. Thus, for applications wherein key size variability is important, MultiDES, one embodiment of the method of the present invention, has a distinctive advantage. Moreover, the key schedule is operative with a variable key setup time.

The advantages of the recommended key schedule include that it generalizes to Feistel block ciphers of different internal structures and block sizes, it causes every subkey bit to be a complex function of master key bits, it allows for a variable length key because each bit individually has significance, and it is more rapid than even the improved key schedule.

Figure 7 is an exemplary illustration of a preferred embodiment which explains how to make and use the form of multiplication portion of the method of the present invention. Preferably, a form of multiplication features the steps as follows.

Preferably, multiply a plurality of bits from plain text-derived-input and a plurality of bits from a plurality of round-dependent subkeys to form a common multiplication product.

Optionally, perform an exclusive-or function on a plurality of bits from plain text derived input and a plurality of bits from plurality of round-dependent subkeys to form a balanced product.

Preferably, perform a combining function on common multiplication product and balanced product to for a pseudo random product. A combining function is typically addition, alternatively, subtraction.

Optionally, fold upper half of first pseudo-random product into lower half of second pseudo random product to form first result. Fold lower half of first pseudo-random product into upper half of second pseudo-random product to form second result.

Optionally, concatenate first result to second result to form folded product.

Preferably, concatenate the lower half of the first folded product with the upper half of the second folded product to form a blended product.

These steps may be repeated, certain steps omitted, and the folding operation modified. The essential constraint is that number of bits flowing out of a step, must equal number of bits flowing to next step.

The method of the present invention uses a carefully planned and specified folding methodology whereby each s-box input is influenced by at least half of input bits. Another embodiment of the method of the present invention is to have each s-box input be influenced, preferably and optionally, by all input bits, thus requiring twice as many key bits per round by repeating multiplication step.

Figure 8 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use the permutation to span multiple blocks portion of the method of the present invention. Preferably, a permutation to span multiple blocks features the characteristic of distributing output of a local scrambler to the inputs of other local scramblers as evenly as possible.

Optionally, the local scrambler is an s-box from the DES prior art.

Optionally, the internal permutation within the scrambler is the P permutation of the DES prior art.

Significant analysis by the inventor revealed that the ideal permutation is one in which the bits are spread out as evenly as possible. To design the scheme, count each bit. Optionally, split the same number of public bits and the same number of private bits in each output. Optionally, split the same number of total bits in each output. Optionally, count a public bit as twice, then distribute the public bits to the neighbors. Optionally, apply the splitting before application of the s-boxes.

The orbital property of the P permutation is defined as the observation that for each s-box, there exists a corresponding s-box. The outputs of a pair of such s-boxes send exactly one bit to each of the s-boxes for the next round, while neither box sends a bit to itself. An extended P permutation is defined as a permutation on groups of s-boxes wherein the orbital property is preserved between (and within) the groups of s-boxes. Where the orbital property is not possible, because the number of outputs is limited, an extended P permutation will distribute the output bits evenly, balancing value of public bits against private bits to break symmetry. Public bits are those repeated by the E expansion. Private bits are those bits used once by the E expansion. A companion execution refers to a wide variety of executions in which a plurality of instances of embodiments of an invention are executed in parallel.

An arithmetic operation refers to a wide variety of ways of combining numbers. One example of an arithmetic operation is a form of multiplication as defined herein. Any method for combining numbers is suitable.

Preferably, the operation of blending is designed based on the observation that each bit of round input in prior-art DES influences four or eight bits in the output of that round (depending on whether the bit is private or public respectively). Due to the property of the prior-art P Permutation, four bits output from an s-box in the round will enter four distinct s-boxes in the next round. The P Permutation in the prior art is constructed so that there exists another s-box whose four bits will enter distinct s-boxes in the next round which are also distinct from those of a specific s-box. Should the reader be familiar with electron orbital and spin, certain metaphors can assist understanding. These properties clearly complement the Feistel structure's property of completeness after exactly four rounds. This observation of the construction of the P Permutation was discovered by the inventor of the present invention. Should it be extended to 5 or 6 blocks, the approach is preferably employed prior to the s-boxes. In another embodiment of the method of the present invention, the blending would take a plurality, such as four, resultant products distributing the bits so that each s-box input of 6 bits would be influenced by maximum possible number of bits from distinct resultant products. Effect of such an operation would be a novel and unobvious extension of the P Permutation to a plurality of blocks.

Figure 9 is an exemplary illustration of a preferred embodiment which explains how to make and use a circuit-based logic-gate implementation of the machine of the present invention. Preferably, define variables, combine them, one, two and three at a time. Write logical expressions using precomputed combinations. Alternatively, the precomputations are reduced by combining only those that are eventually employed. Alternatively , many other combinations of the inputs are possible so long as the operations performed are simple microprocessor instructions. The choice of the combinations to use for each s-box or other type of table entry is substantially determined by the choice of groupings of the variables, of which figure 9 shows an exemplary demonstration. Alternatively, group the variables differently such as one, two, three, four or even all six together..

**HOW TO MAKE BIT-SLICE IMPLEMENTATION LOGIC GATES**

A preferred embodiment of the machine of the present invention employs a logic gate representation. This section describes how the logic gates are generated and used. The machine employs submachines to address the appropriate tasks.

Although the submachines are referred to in this section by name, the actual contents can be readily recreated by a programmer skilled in the art.

A machine "gates" creates the logic gates. The logic gates mimics the S-boxes. In DES, each output bit from an S-box can be viewed as a function of 6 input bits. In the output of the "gates" machine, each output integer is a function of 6 input integers. A structure of "gates" machine including definitions of variables followed by one set per s-box of following: X,Y,A,B,C,D each set to in_sbox[#] as follows. A notation X0 means a value X receives the s box input integer number 0. In a list herein, order is by output integers, each time 'X' reappears a different output integer is referenced.

X 0, Y 3, A4, B 2, C 1, D 5, X 7, Y 8, A 9, B11, C 6, D10,

X12, Y16, A17, B14, C13, D15, X20, Y21, A22, B23, C18, D19,

X25, Y27, A28, B26, C24, D29, X30, Y31, A34, B33, C32, D35,

X38, Y40, A41, B37, C36, D39, X42, Y44, A47, B43, C45, D46.

#include "LOGICDEF.C" (see figure 9)

4 equations, each equation indicates that an output S-box bit is a complicated function of the six input bits X,Y,A,B,C,D.

The functions work as follows: for each output S-box bit position: (from 32 plain-texts) check all 4 possible X,Y possibilities. For each X,Y : check all 4 possible A,B possibilities.

All variable names appearing in the "gates" machine are determined using Polish notation, in which one are two operands are followed by an operation.

Operators: n = not, o = or, p = and, x = exclusive-or. Operands: X, Y, A, B, C, D

Examples: ABon = not (A or B)

CnDx = (not C) exclusive-or D

As mentioned above the "gates" machine builds a machine which is able to employ bit-slice techniques. As input to the "gates" machine are S-boxes. Each S-box has 4 rows, and each row has 16 values each value has a range of 0 - 15. (4 bits).

processing: For each S-box:

For each XY: (S-box row, each row contains 16 values)

For each of 4 output sbox bits:

For each AB .. ...

The object is to determine which of the 16 possible CD values as defined in the CD[16] array applies for each XY, AB combinations.

As an example: S-Box 1

XY=0, AB=0 Top row, 1st 4 entries which are:

14, 4, 13, 1 which in bits are: 1110 0100 1101 0001

represents: ~C;~D ~C;D C:~D C:D

The first output bit of each entry determines out-sbox 0

" second " " " " " " 1

" third " " " " " " 2

" fourth " " " " " " 3

Out-Sbox 0: 1 0 1 0 = 10 CD[10] = "Dn"

1: 1 1 1 0 = 14 CD[14] = "CDpn"

2: 1 0 0 0 = 8 CD[8] = "CDon"

3: 0 0 1 1 = 3 CD[3] = "C" " "

These are indeed the CD values for XYon & (ABon & ..) for the

out sboxes 0, 1, 2, and 3 respectively.

OutSbox[0] =

XYon & (ABon & Dn | ...

Meaning: The output to the 1st S-box bit position depends on:

If neither X nor Y (XYon) and also neither A nor B (ABon)

then the bit is on if not D -- Dn

An operation on 32-bit quantities refers to a wide variety of operations such as arithmetic operations.

Such operations are not intended to include using six bits at a time to perform a table lookup.

A step of calculation of combinations of variables for multiple usage refers to a wide variety of forms of calculation and ways of combinations. Any single or partial step in figure 9 would be suitable, although not limiting.

Figure 10 is an exemplary illustration of an alternative embodiment which explains how to make and use masks derived from DES s-box entries method of the present invention. Table I, a Key Selection Permutation Table, designates which master key bits will be selected for each round subkey. In order to use the table, master key must be circularly rotated by designated amount of the round -shift as noted in the prior art.

Preferably and optionally, a mask will comprise a well balanced number which is, optionally and preferably, a partial permutation and, optionally and preferably, derived from the rows of the s-boxes in DES. Preferably, the mask can be combined with the plain text-derived-round input. Such a step combined with typical row-dependence of the mask to yield a strong mixing function even when initial plain text derived input may not be balanced zeros and ones. Typically, a partial permutation used will depend on the grouping of 32-bits within the plain text derived input.

Figure 11 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use the key schedule portion of the method of the present invention. The top of the figure illustrates inputs of master-key and predetermined initial keys to yield master key derived subkeys. This figure begins to set the stage for subkey-feedback-mode. The middle section of the figure illustrates inputs of master-key and predetermined initial keys to yield master key derived subkeys. These subkeys are used to encrypt in key-generations mode a plain text, which in turn

generates additional subkeys as well as a cipher text. This figure continues to set the stage for subkey-feedback-mode. The lower section of the figure illustrates master key derived subkeys. These subkeys are used to encrypt in key-generations mode a plain text, which in turn generates additional subkeys as well as a cipher text. These additional subkeys are used to encrypt in key-generation mode another plain text, etc. This figure is subkey-feedback-mode.

When employed as a hash function, a preferred embodiment of the machine of the present invention would employ a machine which includes the length of the input into the original input itself. A preferred embodiment of the machine of the present invention for employment as a hash function would apply an integral number of rounds, typically four, of the system. Thereafter, it would generate subkeys from further rounds. These subkeys would be used to influence the next plain text to cipher text transition. (Refer to this mode hereinafter as "subkey chaining mode" or "subkey feedback mode.") This is operative in place of cipher-block-chaining mode. An advantage and object of such a new mode is to avoid known simple relationships between known plain text -- cipher text pairs. Such a known relationship was employed to cause the CBCM mode proposed by IBM to be withdrawn from consideration in the United States of America standard on accepted modes.

A preferred embodiment of the method of the present invention is a stream cipher by employing Output Feedback Mode (using the previous cipher text as the new plain text) using full n-bit feedback. The expected cycle length is $2^{\text{blocksize}}$. An alternative embodiment of the method of the present invention is a stream cipher employing counting mode where the plain text is simply the output of a non-repeating counting mechanism. The next input block would be the previous input block plus one. An object and advantage of employing **TMD** in counter mode is that it allows for accessing the key at an arbitrary distance away, e.g. useful in random access file systems.

In a preferred embodiment of the method of the present invention, Twin TMD is operative employed with two TMD *A* and *B* executions provided. Optionally, employ subkey chaining mode from *A* to *B* and cipher block chaining mode from *B* to a future block in *A*'s sequence.

Folding and blending operations described elsewhere herein apply also to Twin TMD. For brevity, it is not repeated again. The chaining variables could be combined in a Feistel structure with the F function being an entire block encryption optionally using cipher-block-chaining or subkey-chaining.

Figure 12 is a self-explanatory exemplary illustration of a preferred embodiment of the method of the present invention which explains how to make and use the encryption and decryption portion of the method of the present invention. It differs from figure 1 in that it recites fewer optional elements.

Figure 13 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use the key schedule portion of the method of the present invention. Preferably, in the illustrated embodiment, the encryption algorithm is set to use a set of subkeys which are independent of the master key. Optionally, these subkeys are derived from adjacent DES s-box entries along the s-box row.

The figure differs from figure 4 in that it recites fewer elements and generalizes to non-Feistel methods.

In the next figures, a plus sign in a circle is exclusive-or, an empty circle is a form of multiplication, and a plus sign in a box is classical addition.

Figure 14 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use an internal round function portion of the method of the present invention. An exemplary method of folding is shown, wherein the upper half of common multiplication is folded with the lower half. This is simple folding. This is a sample round function for Multi-DES using 64-bit block size. The context of Multi-DES based systems relative to elements present in the prior art is shown.

Figure 15 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use a Feistel structure for Multi-DES portion of the method of the present invention. Showing the Feistel structure approach to this embodiment is used to illustrate that the method of the current invention generalizes to Feistel systems corresponding to the block size chosen. It is illustrated according to a standard representation obvious to one of ordinary skill in the art as described for example in [BiSh93]. The figure shows an example Feistel structure for MultiDES.

As mentioned, the form of multiplication can be applied in non-Feistel structures such as JADE, a system by the inventor of the present invention described at Eurocrypt '97. The folding methodology is applicable to non-Feistel structures with a non-essential example being JADE. The key schedule suggested would apply to any system using subkeys, the number of rounds designated prior to key

extraction will be the first round to reach *completeness*, that each output bit is influenced by each input bit. The examples herein which show generalization beyond the Feistel structure should not be construed to limit. The figure shows an example Multi-DES round function.

See the section above on prior art DES for a detailed explaination of the Feistel structure.

Figure 16 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use a particular form of multiplication portion of the method of the present invention. The form of multiplication illustrated herein includes the steps of multiplying two inputs to yield a product, performing the function of exclusive-or on those two inputs to yield a sum, followed by adding together the product and sum. Another preferred embodiment of the method of the present invention uses common multiplication, folding an upper and lower halves together. Another preferred embodiment of the method of the present invention uses common multiplication folding an upper half of a current execution with a lower half of a companion execution. Additional forms of multiplication apply to the key insertion operation. The figure shows an example multiplication operation in detail, which can be common to many variants of Multi-DES. Reference is made to the section on a form of multiplication above for additional variants.

Figure 17 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use an internal round function portion of the method of the present invention. An exemplary method of folding is shown, wherein *a* folds with *b*, *b* folds with *a*. This is pair wise folding. The figure shows an example round function for **TMD** using two MultiDES encryptions in tandem

Figure 18 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use an internal round function portion of the method of the present invention. An example folding is shown, wherein *a* folds with each *b* and *c*, *b* folds with each *a* and *c*, *c* folds with each *b* and *a*. This is round-robin folding. The figure shows an example Round Function **TMD** using three MultiDES encryptions in tandem

Figure 19 is a self-explanatory exemplary illustration of a preferred embodiment which explains how to make and use an internal round function portion of the method of the present invention. An example folding is shown. The method is *a* folds to *b*, *b* folds to *c*, *c* folds to *d*, and *d* folds to *a*. Refer to the result of *d* folding with *a* as *da*, and the result of *b* folding with *c* as *bc*. Then, preferably

blend *da* with *bc*. An underlying principle is to avoid reusing influence from a given section of plain text derived input wherever there is available other distinct sections. This method is permutation folding. The figure shows an example round function TMD using four MultiDES encryptions in tandem.

The preferred embodiment of the method of the present invention, TMD, can have two, three, four, or more MultiDES rounds run in tandem. A methodology for folding companion round multiplication together to achieve the TMD cipher is shown in figures 14-16. A MultiDES round has a block size of 64 bits and a key size of 64 bits. (Figures 14-15) TMD, using two MultiDES rounds in tandem has a block size of 128 bits and a key size of 128 bits (figures 16-17). TMD, using three MultiDES encryptions in tandem has a block size of 192 bits and a key size of 192 bits (figures 16 and 18). TMD using four tandem rounds has a block size of 256 bits and a key size of 256 bits (figures 16 and 19).

Reference is made to figures 20-26. Step numbers within the set of figures 20-26 are understood to be local to that set of figures unless specifically specified otherwise. Figure 20 is a simplified flowchart illustration of a preferred method for protecting data on a persistent storage medium, such as a hard disk, of a computer, such as a notebook computer. Preferably and optionally, a casual browser does not know that the file system is encrypted. Preferably and optionally, only the user files of the hard disk or other storage device are encrypted. Preferably, the computer continues to decrypt and encrypt files automatically without user involvement whenever a disk read or write occurs.

Briefly, in step 10, the intention to write to a cluster "c" of a hard disk is detected. Typically, encryption is determined on a file-by-file basis, but performed on a cluster-by-cluster basis. Preferably, encryption happens at the time of disk read and writes on the cluster level. Typically, legitimate backup causes work to be perfectly restored whereas illegal backup files remain encrypted. In step 20, information is trapped which is intended to be written to this cluster, for example as entered by a user of the computer. A symmetric cryptosystem is then used to encrypt the information. In step 30, the information is stored in cluster "c" on the persistent medium.

According to another symmetric key ciphering method provided in accordance with a preferred embodiment of the present invention, the following steps are preformed: generation of a key,

encryption of a file using the key, encryption of the key, storage of the encrypted key on a persistant media, typically a hard disk, decryption of the key, and use of the key to decrypt the file.

Typically, the FAT is used to determine the last cluster of a file given a sector within the file.

According to a preferred method for encryption of a storage device, typically a hard disk, the master key for decryption is never present in any form on the hard disk.

Preferably, user-selectable time-out causes requiring reentring password to continue decoding files. Typically, the time out is different for regular use and for idle time.

Figure 21 is a simplified self-explanatory flowchart illustration of a preferred method for protecting confidentiality of information written on notebook computer, the method being constructed and operative in accordance with a preferred embodiment of the present invention. Briefly, in step 100, a pass phrase is provided. The pass phrase typically includes at least 80 to 90 bits of entropy. In step 110, an MD5-MAC key is provided. The MD5-MAC key is typically generated unique to every installation of the method. For example, if software for performing the above MD5-MAC authentication method is installed on a population of hard disks, each hard disk is preferably provided with its own unique key. Typically, this uniqueness of the key is accomplished by cryptographically hashing (e.g. using an MD5 hash or MultiDES-based encryption method operative as a hash) information available on the user's hard disk at time of installation. Optionally, the pass phrase is probabilistically checked for correctness.

Typically, the information which is hashed includes the directory tree. In step 120, the pass phrase is processed using the MD5-MAC key. In step 130, the ciphered pass phrase is partitioned into at least two portions, one of which is the key generation key. In step 140, a file key is generated using the key generation key, as shown in more detail in Figure 22. Preferably, a MD5-MAC authentication method is provided, as shown in figure 21, which can include performing MD5-MAC (described in the above-referenced Menezes document) on a pass phrase and partitioning the result into two 64-bit quantities. Examples of uses for the two 64-bit quantities are described below. Alternatively, use Multi-DES based systems employed as a hash function with 256-bit block size.

Figure 22 is a simplified flowchart illustration of a use of a slightly modified MD5-MAC message authentication code method constructed and operative in accordance with a preferred embodiment of the present invention. Also provided, according to a preferred embodiment of the present invention is

an archetecture for key generation given a sector with 16 DOS directory entries and the number of a specific entry therein. The information is cryptographically mixed to provide a file key. In other words, according to a preferred embodiment of the present invention, a key can be generated by cryptographically mixing a sector having 16 DOS directory entries with the entry from among the 16 entries for which the key is being generated. The cryptographic mixing is typically performed using a symmetric cipher with 64-bit plain text block size and 56 or 64 bit key size. Although, other key-block sizes are possible particularly as provided by MultiDES based systems. Preferably, bytes which participate in the cryptographic mixing are 8 bytes per directory entry starting at 16 Hex, 36 Hex, 56 Hex, etc.

Preferably, the first input to the cryptographic mixing is the specific directory entry and the first directory entry with one playing the role of the key and the other plain-text. Typically, the subsequent input to the cryptographic mixing is the output of the i-1'th mixing ($1<I<17$) and the i'th directory entry with one playing the role of the key and the other of the plaintext. Preferably, the resulting output of the 16th cryptographic mixing is used as a key to encrypt a file. The file key may be encrypted using one of the 64-bit quantities from MD5-MAC or Multi-DES based hash.

Preferably and optionally, the first input to the cryptographic mixing is the specific directory entry and all of the directory sector (512 bytes is 4096 bits per block) with one playing the role of the key and the other plaintext. Optionally, location on disk as calculated in heads, tracks, cylinders, sectors, and offset may be added to the key and/or plaintext before applying Multi-DES based methods to accomplish the cryptographic mixing. Alternatively, the cryptographic mixing is done using a fast parallel bit-wise vector implementation of Multi-DES based systems or DES based systems with a form of multiplication used in place of exclusive or for key insertion within the round function.

In step 200, a sector of a DOS directory and the offset $1<j<17$ of a particular file entry within the sector are provided. Preferably, also the location of the file within the hard disk is also provided (see step 520 of Figure 25 below). A cryptographic key is generated according to the following steps. In step 210, 8 bytes per directory entry are provided, starting at 16 Hex, 36 Hex, 56 Hex, etc to obtain 16 64-bit intermediate keys numbered $0<i<17$. In step 220, these 8 bytes per directory are encrypted with intermediate key $i$ as plaintext and intermediate key $j$ as the key to obtain an intermediate value as ciphertext. Preferably, the location is added to the key $j$ substantially before key $j$ is employed as a

key. In step 230, step 220 is repeated except that the result of step 220 is the key for the encryption to obtain a new intermediate value as ciphertext. In step 240, step 230 is repeated 16 times. In step 260, the resulting intermediate value for $i$=16 is the plaintext and the key generation key from the MD5-MAC or MultiDES-based encryption method operative as a hash as the key for encryption to obtain a file key as ciphertext. Optionally, differential time between keypresses or disk latency time or the contents of keystrokes or contents of disk reads are used to seed a random number generator.

Figure 23 is a simplified self-explanatory flowchart illustration of a preferred method for generation of file keys forming a part of the method of figure 22, using contents of DOS directory entries as plain texts and keys to generate a file key. In step 300, a symmetric cipher key is generated, for example according to Figure 22. In step 310, a file or directory is encrypted with a symmetric cipher, for example with MultiDES-based encryption methods, such as that shown in Figure 26. In step 320, the file key is encrypted as plaintext using a key protection key, typically generated according to Figure 21, as key with a symmetric cipher to obtain a protected file key. Alternatively, the file key is generated by employing information available in the sector of the directory of the file, using MultiDES-based encryption methods, employing the specific file entry as the key and the remaining part of the sector as the plaintext. In step 330, the protected file key is stored in a conveniently located portion of the disk, for example in the last bytes of the last cluster allocated to the file.

Figure 24 is a simplified self-explanatory flowchart illustration of preferred method for performing an encryption of a file using the method of figure 22 to generate file keys and the output of the method of figure 21 to protect the file key. In step 400, a symmetric cipher key is generated, typically using Figure 22, or using MultiDES-based encryption methods as mentioned above. In step 410, a file or directory is encrypted with a symmetric cipher and the key is stored, for example according to Figure 23. In step 420, a key protection key is provided, typically generated according to Figure 21 or using MultiDES-based encryption methods effective as a hash function. In step 430, the protected file key is retrieved from a conveniently located portion of the disk, substantially as previously described. In step 440, the protected file key is decrypted as ciphertext using a key protection key, for example generated according to Figure 21 or using MultiDES-based encryption methods effective as a hash function, as key with a symmetric cipher to obtain a file key. In step 450,

the file is decrypted by using the file key as the key by using conventional methods, or alternatively according to Figure 26.

Figure 25 is a simplified self-explanatory flowchart illustration of preferred method for performing an encryption of a file on a sector by sector basis using unique information based on the location on the particular hard disk and cipher-block-chaining within the sector. According to a preferred symmetric key ciphering method provided in accordance with the present invention, ciphering proceeds as follows: given a key, and a sector number of data to be encrypted, encryption is carried out typically using the location serial number as an initial vector. Thus, preferably employ subkey-chaining-mode together with bit-slice vector implementation to maximize block size for Multi-DES based method. According to another file encryption method provided in accordance with a preferred embodiment of the present invention, a symmetric cipher key is generated, a file is encrypted and a protected file key is stored. The protected file key is typically stored in a conveniently locatable place on the disk, typically in the last bytes of the last cluster allocated to the file.

A preferred method for protecting hard disks uses an available attribute bit from the attribute byte, typically bit 6, to indicate whether or not to encrypt. Preferably, there is a default as whether or not to encrypt, the default being, for example, to encrypt. Preferably, each file handle, upon opening the file, is associated with a bit which indicates whether or not to encrypt the contents of the file.

Typically, the association is a simple index into a 256-byte table.

In step 500, a key is provided, for example according to Figure 22 or using MultiDES-based encryption methods effective as a hash function, as key with a symmetric cipher to obtain a file key. In step 510, a sector number of the data to be encrypted is provided. In step 520, a location serial number is obtained by deriving sector number information which is unique to the presently installed hard disk and current location, such as hard drive number, cylinder number, sector number, and number of the read/write heads. In step 530, a sector is partioned according to the symmetric cipher block size into plaintext blocks, for example according to MultiDES-based methods. In step 540, the sector is encrypted with cipher-block-chaining or sub-key-block-chaining mode of the methods of the present invention (for example as shown in figure 11), by using conventional methods according to the location serial number as the initial vector.

Figure 26 is a simplified self-explanatory flowchart illustration of preferred method for performing the method of figure 25 wherein the encryption is fast parallel bit-wise vector implementation of DES with a form of multiplication substituted for exclusive or when combining the subkey with the plaintext derived input, such as MultiDES. The symmetric cipher is typically a fast parallel bit-wise vector implementation of DES using a form of multiplication for key insertion. The size of the bit-wise vector is preferably a multiple of 8 such as 16, 32, or 64. MultiDES is operative a sector at a time as well as a cluster at a time.

The above methods and systems are useful for many storage devices such as hard disks and such as the hard disk of a portable typically notebook computers in particular.

It is understood that in figures 27-36, wherever addition or multiplication is used, a form of multiplication may be substituted. Likewise, parallel execution(s) may be combined using techniques of folding and/or blending. In the figures 27-36, the symbol +, not encircled, indicates standard addition. Step numbers within the set of figures 27-36 are understood to be local to that set of figures unless specifically specified otherwise. The symbol +, encircled, indicates an exclusive-or operation. The symbol + in square indicates standard addition.

Reference is now made to figure 27 which is a simplified flowchart illustration of a DES encryption method constructed and operative in accordance with a preferred embodiment of the present invention. A suitable initial permutation (e.g. for step 10) and a suitable final permutation (e.g. for step 30 as well as for step 320) and a suitable DES key schedule (e.g. for step 50) are all described in Biham and Shamir's Appendix A, "Description of DES" and/or in the Glossary of the above-referenced Biham and Shamir publication.

To obtain an inverse of the P permutation, conventional methods may be used to compute an inverse of the P permutation described in Biham and Shamir's Appendix A and/or glossary of the Biham and Shamir publication.

The subkey table generated in step 50 may, for example, be stored on a hard disk. It is stressed again that a form of multiplication specifically includes common addition as a possible form.

Figure 28 is a simplified flowchart illustration of a first preferred method for performing an n'th DES round forming part of the method of previous figure, using addition to combine subkey with

plain text derived input of the method of the previous figure (step 80). The method of the current figure uses a form of multiplication to combine subkey with plain text derived input.

Figure 29 is a simplified flowchart illustration of a second preferred method for performing an n'th DES round (step 80) employing a form of multiplication part of the method of figure 27. The method of the current figure, like the previous figure, uses a form of multiplication to combine a subkey with plain text derived input. However, in the previous figure, only a single plain text is typically encrypted at a time whereas in the current figure, a plurality 'I' of plain texts, such as 'I' = 32 plain texts, each including I bits (typically I=64) are encrypted simultaneously. Typically, the DES encryption method of figure 27 is repeated I times and the i'th performance (I=1, ..., I) of the DES encryption method of figure 27 encrypts an i'th bit of each of the plain texts. Optionally, third and fourth permutations may be used which respectively replace the first and second permutations of steps 40 and 120 respectively. The third permutation is defined by associating the i'th bit of the t'th plain text derived input. The fourth permutation is defined by associating the i'th bit of the t'th ciphered text (t=1, ..., T) with the t'th bit of the i'th final round output. If a plurality of plain texts are encrypted simultaneously using exclusive-or to combine subkey with plain text derived input, then the encryption output is no different than it would be if the plurality of plain texts were to be encrypted one by one, the only advantage of simultaneous encryptions being speed. However, if as is shown in the previous figure, the plurality of plain texts are encrypted simultaneously using a form of multiplication (not exclusive-or or another bit-wise operation) to combine subkey with plain text derived input, then the encryption output is different than it would be if the plurality of plain texts were to be encrypted one by one. It is appreciated that because the results of the encryption method of the current figure are different than the results of conventional DES, the initial and final permutations of DES may be skipped, to increase speed. When step 310 is performed for a first round of DES encryption (n=1 in figure 27) the plain text derived input typically comprises the plain text itself. When step 310 is performed for a subsequent round of DES encryption (n>1 in figure 27) the plain text derived input typically comprises the output sequence of 64 integers generated in step 370 of the previous round n-1. The expansion table used in the current figure, step 310 is typically the same expansion table used in figure 28, step 140. Step 350 may be performed using any of possible logic gate configurations described herein and others. In the current figure, the length of

each subkey-derived integer and each plain text derived integer may be any suitable length such as 8 bits, 16 bits, 32 bits or 64 bits.

Figure 30 is a simplified flowchart illustration of a modification of figure 28 in which first and second permutations and mapping are employed to perform the DES round; useful when steps 10 - 30 are employed. In the method of the current figure, the mapping generated in step 30 is employed to perform a DES round.

Figure 31 is a simplified flowchart illustration of a third preferred method for performing an n'th DES round forming part of the method of figure 27, wherein subkeys are combined with plain text derived input using a form of multiplication as shown. A preferred method of the current figure is combination of s-boxes, permutation and expansion into a single table look-up.

Figure 32 is a simplified flowchart illustration of a DES encryption method constructed and operative in accordance with another preferred embodiment of the present invention.

Figure 33 is a simplified flowchart illustration of a fourth preferred method for performing an n'th DES round forming part of the method of figure 32, using multiplication to combine subkey with plain text derived input.

Figure 34 a simplified flowchart illustration of a fifth preferred method for performing an n'th DES round forming part of the method of figure 32.

Figure 35 a simplified flowchart illustration of a modification of figure 33 in which first and second permutations and mapping are employed to perform the DES round.

Figure 36 is a simplified flowchart illustration of a sixth preferred method for performing an n'th DES round forming part of the method of figure 32.

Appendices include a description of research based on findings which indicate that replacing the exclusive-or operation with an addition operation, with the F function described by Biham and Shamir, does not always yield a weaker cryptosystem, contrary to the teachings of Biham and Shamir in section 4.5.3.1 of Chapter 4 of the above-referenced Biham-Shamir publication. The research findings described in appendices also indicate that replacement of exclusive or within the F function by common multiplication with final carry discarded is, in certain situations, stronger than conventional DES methods. The research findings also suggest that replacement of exclusive-or

within the F function by multiplication over a ring is preferable to replacement of the same by common multiplication with final carry discarded.

The method of the present invention provides a rapid, simple, and secure means for controlling a microprocessor to effect symmetric message authentication, one-way hashing with or without a key, and a symmetric block cipher.

Many other variations are possible, for example, the expansion mapping is unnecessary when a form of multiplication is used for key insertion. Other variations are possible, for example, the key insertion and folding operations can be applied to a variety of ciphers to yield improved block size regardless of whether the Feistel structure or a totally different construction is used.

Other variations are possible, for example, the key insertion and folding operations can be applied to a cipher whose block length is any arbitrary amount **shorter** than the designated block length by replacing the influence of plain text derived input by additional subkeys in each round. For example, to shorten a 64-bit block cipher to 48-bits only simply encrypt normally, but at the start of each round where plain text derived input is required use 48-bits only and use additional 16-bits of subkey that round. The key schedule would require more rounds at key set-up time to effectively generate the additional subkey bits for each round.

Other variations are possible, for example, the key insertion and folding operations can be applied to a cipher for whom every s-box is influenced by every plain text derived input bit in a round. In place of the blending operation described, use form of multiplication for key insertion again on a distinct set of subkeys. These results can be blended between the first and third s-box inputs as well as the second and fourth s-box inputs. Blending takes half of the output from one of the arguments and the other half from the other.

Other variations are possible, for example, the key insertion and folding operations can be modified so as to use any group operation or operation which combines a few group operations. For example, the folding can be done using addition, subtraction, or even modular multiplication or division.

Other variations are possible, for example, the key insertion and folding operations can be combined with a bit slice implementation where the size may be chosen based on considerations of existence of Fermat primes.

Likewise, common multiplication can be used as an expansion operation instead of the standard E mapping. Multiplication of a 32 bit subkey by a 32 bit plain text derived input, yields a 64 bit quantity. We may discard the upper and lower 8 bits of the result, leaving us with 48 bits which can be fed into the S-boxes. Likewise, the expansion mapping could be accomplished after the key insertion operation. This has the advantage and object of simplifying the round function and causing the bits entering the s-box to depend on a plurality of plain text derived input bits as distinct from the prior art wherein the dependence is on a single plain text derived input bit.

Execution of two operations of block or stream encryption in parallel can employ common multiplication with exclusive-or to fold the upper half of the result of the multiplication into the lower half of the companion execution. This is referred to as MultiDES based systems, one embodiment of the method of the present invention. An object and advantage is to extend the key-block length by causing mutual influence of plain text derived input bits on the other respective round output.

If the two operations are MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention, encryptions running in parallel, similar folding techniques can be applied in parallel yielding a 1024-2048-4096 bit block cipher called MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention. An object and advantage is to extend the key-block length by causing mutual interference of plain text derived input bits on the other respective round output. Moreover, a five-fold speed increase is achieved relative to the embodiment with non-bit-slice s-boxes.

The bit-slice implementation does not need to encrypt exactly 64 plain texts at once. Rather, preferably and optionally, encrypt 4, 8, 16, or 32 at a time. This enables the group operation of multiplication using a Fermat prime to combine 16-bit subkey with 16-bit plain text derived input. Naturally, simultaneous encryption of 2, 4, and 8 plain texts typically use multiplication over a field modulo a Fermat prime. Multiplication over a field modulo a Fermat prime can be improved relative to the all-zero key by treating zero as "-1" over the field. An object and advantage of modular multiplication is due to being a permutation, it is known that all bits in the domain and range are used.

In fact, use of common multiplication using some method such as exclusive-or to fold the upper and lower halves together, or multiplication over a ring (or a field) should yield similar results.

Unlike Biham's solution, this is not plug-compatible with prior art DES. Rather, it causes each of the 16-bit plain text derived inputs in a given round to influence at least one bit in every other simultaneous encryption. Because there exists a given 16-bit input, for a fixed key, yielding any particular desired output bit combination. Thus, Shannon's criteria of diffusion and confusion are better satisfied. Multiplication over a field defined by a Fermat prime is no longer computationally expensive in the Intel microprocessor architectures. Our method would extend to any method of multiplication which could be simply expressed as a combination of the resulting two input-sized results from common multiplication. The group operation chosen within a round to combine the subkey with plain text-derived-input need not be constant from round to round. The method of the present invention may be cascaded, used before or after known or to be invented methods.

Advantages and objects of the bit-slice ramification of method of the current embodiment of the present invention include additional speed, additional block size, effective hardware implementations, encryption block size matching that of public-key algorithms such as RSA, convenient stream cipher, and powerful hash function.

One advantage and object of the method of the present invention is increased speed. MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention, achieve block throughput rates averaging about five times as fast as prior art DES. This improvement is achieved without reducing the number of rounds.

Another advantage and object of the method of the present invention is increased block size.

The huge block size ranging from 512 bits to 4096 bits breaks up local patterns effectively and depends on every single bit of key and/or plain text input.

Another advantage and object of the method of the present invention is effective hardware implementations.

The approach is suitable for use on many computers not limited to 64-bit microprocessors such as the DEC Alpha, or on 32-bit microprocessors as the Intel Pentium. Likewise, an efficient example implementation using logic gates has been accomplished.

Another advantage and object of the method of the present invention is encryption block size matching that of public key systems. For the first time, a secure symmetric system achieves the same block size as RSA.

Thus, it would be a natural partner to RSA in new protocols. For example:

1. signatures based on a partial or folded output of MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention.

2. zero-knowledge identification in which partial outputs of MultiDES based systems with bit-slice implementation, one embodiment of the method of the present invention are shown.

3. digital cash in which spending a coin reveals a key and partial output

double-spending would be caught when more output is revealed.

Another advantage and object of the method of the present invention is an effective stream cipher and hash function. The ability to effectively mix significant chunks of data allows for a natural application as a pseudo-random number generator to be used as part of a stream cipher. Likewise, huge inputs are rapidly hashed to the desired size.

An advantage and object of the archetecture of the present invention is that sleep to disk causes encryption of memory being written to disk and/or erasure of the master key in memory prior to writing to disk.

Another advantage and object of the archetecture of the present invention is that an enemy who captures of a computer which is powered off (or in smart-sleep state where memory is written to disk) gains nothing except the encrypted data.

Another advantage and object of the archetecture is that recovering the data requires either knowledge of the pass phrase or equivalent of breaking an accepted or patented encryption method.

Another advantage and object of the archetecture is that typically, identical files encrypted under the same key do not yield even the same initial encrypted block.

Another advantage and object of the archetecture is that typically, files which are not cryptographically sensative are not automatically encrypted.

Another advantage and object of the archetecture is that typically, user files and newly created files are automatically encrypted.

Another advantage and object of the architecture is that typically, encrypted and plaintext files co-exist on all but the most security intensive systems.

Another optional advantage of the architecture is that it is not obvious that encryption has been used, except for used hard disk space for which no files lay claim.

Another application and object of the method of the present invention being in a wide variety of applications including fast communication links and local applications e.g. for confidentiality and authentication purposes, particularly including automatic, background encryption of hard disks of notebook computers, preferably on a file-by-file basis; encryption of file names on a storage medium; encryption of file contents, encryption of file names of those files and omission of information regarding those files from file directory listings; trapping all READs and WRITEs to the disk either on the DOS level or on the BIOS level; trapping any "sleep" mode writing to a disk of a notebook or desktop computer; cluster-by-cluster encryption; sector by sector encryption; use of bits in an attribute byte for deciding where or not to encrypt a file; use of cipher block or subkey generation chaining mode over the largest block read or written by the chosen operating system as a single unit (typically a sector, cluster or track), employing sector number, track number, head number, cylinder number, cluster number, disk drive serial number, and any other available information to characterise a present location within a specific hard disk; and encryption of a cipher key and placing it in a location within a hard disk which is easily addressable given the cluster number of a cluster within a file. For example, the easily addressable location may be the last bytes of a last cluster in file which contains a cluster whose number is given or in the directory or cluster allocation information related to file.

Another suitable method for implementing the method of the present invention involves optimisation of 32-bit parallelism and 32-bit registers running in protected mode or optimisation of 16-bit parallelism and 16-bit registers running in real mode or optimisation of 32-bit parallelism and 32-bit registers running in real mode with 32-bit op-codes or optimisation of 64-bit parallelism and 64-bit registers running in real mode using a floating point unit to perform 64-bit arithmetic operations. Preferably, each input register to any of arithmetic operations shown and described in figures 27-36 is fully utilised but carries are preferably ignored as necessary depending on size of available registers.

## WORK ON PRESENT INVENTION: IMPLEMENTATION DETAILS

Various previously described features of the present invention were tested in actual implementations of cryptographic software for performing the methods of the present invention. The described techniques were applied to improve the preferred method of the present invention, yielding

speed of implementation of 16,000 bytes per second on a Pentium 120 Mhz machine. Some approaches to enhance speed included the following: calculating the 16 sub-keys in advance, instead of for each plain text. In calculating the sub-keys, the method combined the: initial key permutation, key shift per round, compression permutation and output a 48 by 16 table, relating position of compression bit with bit position of original key. Further, the method combined the P-box permutation, the initial and final permutations, and the Expansion Permutation, into the H and F tables, causing the P-box permutation to 'disappear' . Thus, a preferred method of the present invention was implemented as follows: The subkey bit positions (by round) were calculated in advance. The plain text was permuted by an F[] table, and then split into f1[] and f2[] with f2[] being used in a round. In each round, the H[] table re-ordered f2[] Reordered bits were then applied the function exclusive-or with the subkey. The S-boxes were viewed as an array of 64 values. Then the function exclusive-or value was applied to the element number value in that position in the array of the S-box output. f3[] was then applied to the function exclusive-or result of f1[] and the S-box output. At the end of each round (including the last), f2[] was copied into f1[] and f3[] was copied into f2[]. After the 16 rounds, a T[] tables determined the cipher text bit positions from f1[] and f2[]. A preferred embodiment of the machine of the present invention was enhanced with programming techniques to speed up the program. New speed recorded was 34,000 bytes per second on a Pentium 120 Mhz machine. Techniques employed included using integer registers, and using an 'nbit' 2 dimensional table, where the row number was the numerical value, and the row itself was the bit representation of that value. In going from nibbles to bits, the method copied the nbit[row] where row was the value of the nibble. (16 rows in all). Additional steps included changing f1[], f2[] and f3[] to a 2-dimensional table, f[][] where the row value was represented by variables f0, f1, f2. initialized as 0,1,2 respectively. From round to round 1 modulo 3 was added to these values. The steps in each round were now as follows: the nbit's at row f1[] were copied to an array called 'fbit' For each S-box input: used H[] value to choose a bit from fbit; used Left Shift and exclusive-or to accumulate k Exclusive-or k with the sub-key [round, S-box]. f[f2] = S-box[k] ^ f[f0]. Rotated: m–f0, f0–f1, f1–f2, f2–m even at the end of the last round.

The method used also applied bit-slice methodology enabled encrypting 32 plain texts at a time. The speed was 506,000 bytes per second on a Pentium 120 Mhz machine. 257,000 bytes per second

included the bit-splitting that proved that this was in fact an implementation of DES. An implementation included: the 64-bit key was converted to the 16 round sub-keys, except that each bit was expanded to an integer of 32 bits. Therefore, each subkey contained 48 integers, each integer was either all zeros or all ones.

A machine started operation by reading 32 blocks of 64-bit plain texts for a total of 64 integers were at a time. Every 2 integers represented one plain text. The data was optionally re-arranged as follows. Every plain text has 64 bit position. Each bit went into a separate integer. The bits of the first plain text went into the high order position of each integer. The bits of the second plain text went into the next highest order and so on. As a result each output integer represents one bit position of every plain text. The implementation follows the ideas of combining permutations. The plain text was permuted by an F[] table, and then split into f[f0] and f[f1] with f[f1] to be applied the function exclusive-or with the sub-key. In each round, the H[] table re-orders f[f1]. The reordered bits were then applied the function exclusive-or with the subkey, with the result going into an array of 48 integers.

The data went into the logic_gates() routine, which was described in detail in a description of figure 9. The logic-gates mimicked the S-boxes by viewing each output integer as a function of 6 input integers. f[f2] = f[f0] applied function exclusive-or with the logic gate output. f0, f1, and f2 values rotate (m=f0; f0=f1; f1=f2; f2=m) even at the end of the last round. After the 16 rounds, a T[] tables determines the cipher text bit positions from f[f0] and f[f1]. Finally, the bits were re-arranged, in a procedure reverse of the original: Each cipher text contains one bit position of each of 64 integers.

The Bit Split and the Undo Split were essentially an input and final permutation, and can be omitted without compromising security. They were retained to prove that this program was in fact an implementation of DES. The speed of a preferred embodiment of the present invention for executing triple-DES was 171,000 bytes per second on a Pentium 120 Mhz machine. 127,000 bytes per second included the bit-splitting. Triple-DES was 1/3 as fast as DES without bit splitting and ½ as fast as DES with bit splitting. A preferred embodiment of the machine of the present invention for triple-DES used 3 keys. It encrypted with the first, decrypts with the second and encrypts again with the third. This embodiment was tested by setting the second and third keys to be identical. The program

was almost identical to that of a previous implementation described above except that it used 48 rounds. Subkey generation pre-calculation stored the sub-keys in decryption phase in reverse order. Created 160,000 encryption in 44 seconds. Plain texts were generated by a random generator, employed DES on plain texts to yield cipher tests. Timed random generator as generating 7 million bytes if data in 32 seconds, which came out to 6 seconds for amount of data uses. Therefore, the DES operations took 38 seconds. Results were compared to a test bed of data. Called the random generator only one for the initial plain text. Called DES using the output of the previous call as input to the current call. The mechanisms for timing was constructed as follows. For this particular test the software did not write out to disk, and the clock was started right before the first call to DES. At end of the program, calculated speed by time elapsed to encrypt 10,000 * 32 blocks of data.

A reduced P permutation is a permutation substantially similar to those shown in figure 8.

The method of the present invention relates to using a form of multiplication as the key insertion operation and related folding methodologies useful to form a shorter input length keyed hash function. Another method of the present invention employs bit-slice methods. The preferred embodiment of the method of the present invention is rapid, simple and can be shown superior to prior art DES which has faced the tests of time. The method of the present invention achieves a 256-bit input size, yielding a 128-bit output in the preferred embodiment.

It is appreciated that various features of the invention which are, for clarity, described in the contexts of separate embodiments may also be provided in combination in a single embodiment. Conversely, various features of the invention which are for brevity, described in the context of a single embodiment may also be provided separately or in any suitable subcombination.

It is appreciated that various features of the invention which are, for clarity described in the contexts of separate embodiments may also be provided in combination in a single embodiment. Conversely, various features of the invention which are for brevity described in the context of a single embodiment may also be provided separately or in any suitable subcombination.

It is appreciated that the software components of the present invention may, if desired, be implemented in ROM (read only memory) form. The software components may, generally, be implemented in hardware, if desired, using conventional techniques.

While the above description contains many details, these should not be construed as limitations on the scope of the method of the present invention, but rather as an exemplification of at least one preferred embodiment thereof. Accordingly, the scope of the present invention should be determined not only by the embodiment(s) illustrated including appendices, but also by the appended claims and their legal equivalents.

APPENDIX

## THEORY OF OPERATION

### BACKGROUND FOR THEORY

*Multi-DES* is a new cipher based on standard DES with the same modification as SuperDES without the bit-slice implementation. For analysis, we defined a variant of common multiplication wherein the upper half of the result is discarded. Differential cryptanalysis using the XOR as the differential yielded so many restrictions on the key as to make the number of possible characteristics insufficient to recover all possible keys. The best characteristics which we were able to find multiplied by the likelihood of a key satisfying it was approximately the cost of exhaustive search. Thus MultiDES is stronger than DES.

Continuing our analysis, we defined a variant wherein the multiplication is done over a field. One example of a field is that generated by multiplication modulo $2^n+1$ (when such a number is prime). Thus, we changed the group differential from XOR to ratio over a field. We attempted to build differential distribution tables for the behavior of the input and output ratio over the field for the s-boxes. Likewise, we needed such difference tables for the combination of the P permutation composed with the E expansion. This further decreased the chance of any successful differential. Moreover, connection from one round to the next of the characteristic cost additional probability.

Because these variants, e.g. where n=16, simply the tables defined were large, we, for analysis, defined a reduced variant from 12 to 8 bits (with the expansion mapping defined accordingly). Likewise, because attack on 16 rounds was difficult, chose to attack two rounds using heavily the properties of the Feistel structure of DES.

## ATTEMPTED CRYPTANALYSIS of METHOD AND MACHINE

To effectively evaluate the potential of the **TMD** cipher, we attacked it with known cryptanalytic methods. Since the **TMD** cipher is a tandem application of two or more MultiDES encryptions we began our analysis by studying Differential Cryptanalysis of MultiDES.

MultiDES replaces the internal XOR in the F round of DES with common multiplication. Upon study of differential cryptanalysis, we found that substitution of common multiplication for XOR in the F function of DES yields a cryptosystem which is different from DES. We investigate the behavior of the MultiDES input difference **K(E')**:

$$K(E') = (K \cdot E) + (K \cdot E^*) \qquad \text{.....(1)}$$

Where :

$$E' = E + E^* \qquad \text{..... (2)}$$

Here **K** is a Key, **E'** is the XOR input difference (as used in differential cryptanalysis of DES), **E** and **E*** are input plaintexts. **E'**, **E** and **E*** are all valid expanded texts which obey the e-expansion.

In differential cryptanalysis we seek input differences to the substitution boxes which form the best iterative characteristics, those having the highest relative probability. Using some of these best choices for characteristics we will show MultiDES is **stronger** than DES in a differential cryptanalysis attack based on classical difference distributions. In addition, we will show that limitations are placed on the key space for compliance with the best characteristics chosen; consequently, weak keys, those that comply with high probability characteristics and allow a differential cryptanlysis attack,

are few.

Limitations are placed on the possible bit patterns for a valid expanded text (**E, E***). When a 32 bit text is expanded into a 48 bit text (12 nibbles in Hex notation) nibbles 2, 5, 8, and 11 of the expanded text must be symmetric, in the sense that, the two left bits of the nibble must be identical to the two right bits of the nibble. This is true, since these particular nibbles have bits that are shared by adjacent substitution boxes. This limits the Hex value of these nibbles to 0, 5, A, or F. The remaining nibbles must have adjacent nibble symmetries, in the sense that, the two rightmost bits of nibble i (i=1, 3, 4, 6, 7, 9, 10, 12) must be identical to the two leftmost bits of nibble i+1. Therefore, the expanded text pattern (**E, E***) associated with a given substitution box (before key multiplication) is limited according to the particular nibbles it contains. The result of the common multiplication of the expanded text and the subkey, **K(E')**, is not required to be a valid e-expansion entity. This iterative characteristic can be formed, for a given substitution box, from input differences which yield a **zero** output difference with a **high probability**.

In particular, for best results, we consider input differences that affect only **isolated** substitution boxes.

In order to meet the isolated substitution box constraint, we have limitations on the bit patterns of the two nibbles of the input difference. It is noted for convenience, that the nibbles of the **input difference** for a particular substitution box are related to the nibbles that enter the particular substitution box in the following way. For even numbered substitution

boxes (2, 4, 6, or 8), the input difference nibbles are mapped directly from the nibbles entering the substitution box (the right nibble of the input difference is the right nibble that entered that particular substitution box and the left nibble of the input difference is the left nibble that entered that substitution box ).For odd numbered substitution boxes (1, 3, 5, or 7), the bits of the input difference nibbles are **not** mapped directly to the bits of the nibbles entering that substitution box; but rather, the right nibble of the input difference is composed of the two leftmost bits of the right nibble entering that substitution box with the two rightmost bits of the left nibble entering that substitution box, while the left nibble of the input difference, is composed of the two leftmost bits of the left nibble entering that substitution box with the two rightmost bits of a nibble from the previous substitution box. For input differences entering an even or an odd substitution box the leftmost two bits of the left nibble (of the input difference) must be zero, since they are in a previous substitution box. In addition; since, both **E** and **E*** must obey the e-expansion and also not affect neighboring substitution boxes, they both have the two leftmost bits of their left nibbles and the two rightmost bits of their right nibbles zero. Moreover, we note that the two rightmost bits of an input pattern are conserved over key multiplication (i.e., the two rightmost bits of the input pattern **K(E')** which are obtained after key multiplication with **E** and **E*** remain zero, as they were in **E, E***, irrespective of the key).Therefore, for any substitution box, both the two leftmost bits of the left nibble and the two rightmost bits of the right nibble, of the input difference, must be zero. Consequently, the input difference **K(E')** is limited to the Hex values: 0, 04, 08, 0C, 10, 14, 18, 1C, 20, 24, 28, 2C, 30, 34, 38, or 3C. For example, in

substitution box 4, an input difference of 28 Hex meets all constraints and is a candidate from which we can obtain a high probability iterative characteristic; since,

$$\text{Prob}(28 \rightarrow 0) \text{ is } 16/64 = 1/4.$$

(It is noted that the **only** high probability (1/4) entry in any substitution box Difference Distribution Table which also obeys the input difference constraints listed above is the input difference $28_x$ in substitution box 4).

In order to fully understand the behavior of Eq. (1), we first studied a version of MultiDES in which common multiplication is performed in each substitution box independently. The input to a particular substitution box is therefore only a product of the expanded text associated with that substitution box (a six bit entity) and the bits of subkey associated with that particular substitution box (six bits of subkey). The result of the common multiplication of two six bit entities is an eleven bit entity. We discard the five bits of the upper half (left half) of the result of the common multiplication; and, retain only the six bits of the lower half (right half), and use these six bits as the input to the substitution box.

The input difference of Eq. (1) is not generally conserved with respect to the XOR input difference Eq.(2). For the particular case of a zero input XOR (**E'** = 0), Eqs.(1) and (2) are equal, and the input difference for MultiDES is identical to the XOR case. This case is of no practical use in differential cryptanalysis since all keys are equally probable. We now introduce a Lemma which will assist in the selection of characteristics.

**Lemma**: If **r** is the bit location ( counted from the right of the bit pattern)

of the first non-zero bit in the XOR input difference **E'**, and s the bit position

(counted from the right of the bit pattern) of the first non-zero bit in the key **K**, then the first non-zero bit in the MultiDES input difference **K(E')** is located at bit position **r + s - 1** (counted form the right of the bit pattern of **K(E')**).

Corollary : **E'** can only have a unique non-zero bit.

The corollary results from the observation that Eq. (1) holds together with the conditions of the Lemma, if and only if, when multiplying both **E** and **E*** by a constant (the key) and XORing the result, an input difference **K(E')** is obtained which is a shifted bit pattern of the key. (For input XOR's, **E'**, having more than one non-zero bit, **K(E')** is not a shifted bit pattern of the key.)

In the case at point, input difference to substitution box 4 having Hex bit pattern 28 (101000), $r + s - 1 = 4$. This means there are several choices for characteristics:

$$r = 4 \quad s = 1,$$

$$r = 3 \quad s = 2.$$

This constrains the choices of **E'** and the corresponding keys which can give a given **K(E')**.

These r values (together with the corollary) limit the input XOR, **E'**, into substitution box 4, to 08, and 04 (Hex).

These s values give us our first constraints on the subkey bits entering substitution box 4. Allowable keys have bit patterns of xxxxx1 or xxxx10 (where x can be arbitrarily 0 or 1). We now show additional constraints on

the key bits entering substitution box 4.

Case r = 4, s = 1:

Assume the key, K, has piece entering substitution box 4 with bits $k_1$ $k_2$ $k_3$ $k_4$ $k_5$ $k_6$. (These bits are those entering substitution box 4 and are numbered for convenience with respect to this substitution box, but they are really bits 19-24 of the subkey, as counted from left to right or bits 25 to 29 when counted from right to left.)

Since s = 1, $k_6$ = 1.

Assume E has bits a b c d f g and E* has bits a* b* c* d* f* g* ,entering substitution box 4.

To comply with the Lemma and its corollary, E'= 000008000000 (Hex), and therefore, c must compliment c*.

Without loss of generality we can write: a = a*, b =b*, c <> c*, d = d*, f = f* and

g = g*. We can also set c = 1, causing c* = 0.

Consider the expression:

$$E^* \cdot K = (E \cdot K) + (E' \cdot K)$$

.....(3)

We evaluate the right hand side of Eq. (3).

**(E' · K) = (0 0 1 0 0 0) · ($k_1$ $k_2$ $k_3$ $k_4$ $k_5$ 1)** whose right half pattern is **$k_4$ $k_5$ 1 0 0 0.**

**(E · K)** is unknown and assumed to have bit pattern: **A B C D F G.**

**(E* · K)** is given Eq. (3) and evaluated :

75

| | | | | | | |
|---|---|---|---|---|---|---|
| | A | B | C | D | F | G |
| + | | | | | | |
| | $k_4$ | $k_5$ | 1 | 0 | 0 | 0 |
| | A* | B* | C* | D | F | G |

The following conditions are automatically valid: **C <> C*, D = D*, F = F*, G = G*.**

In addition; since, K(E') (Eq. (1)) was set at (Hex) 28, (bit pattern 1 0 1 0 0 0),

**A <> A***, and **B = B***.

We now examine carry arithmetic involving the bits **B, B*** ; and **A, A*** of (E · K) and (E* · K).

Carry arithmetic:

We are going to calculate carry in the addition: (E · K) + (E' · K). Suppose we have sum U+V=W in binary signature with corresponding bits $u_i$ , $v_i$ and $w_i$ in column I

(i = 0, 1, ..., n). The preliminary value of sum in column i+1 is $Sum_{i+1} = u_{i+1} + v_{i+1} + carry_i$. The real value of sum in column i+1 is $w_i+1 = Sum_{i+1}$ (mod 2). Since $u_0+v_0 <= 1+1=2$ we have $carry_0 <= 1$. Suppose that $0 <= carry_i <= 1$. Then $Sum_{i+1}= u_{i+1}+v_{i+1}+ carry_i <= 3$. Therefore $0 <= carry_{i+1}= (Sum_{i+1}- w_{i+1}) / 2 <=1$.

So always $0 <= carry_i <=1$. $w_i= Sum_i- 2\ carry_i$.

In general:

$$w_i = u_i + v_i + carry_{i+1} - 2\ carry_i.$$

.....(4)

We investigate the carry arithmetic for our conditions $B = B^*$, $A <> A^*$:

$$B^* = B + k_5 + carry_C - 2\,carry_B$$

..... (5)

Now $\underline{B^* = B}$, if and only if, $k_5 + carry_C - 2\,carry_B = 0 \pmod 2$.

If $carry_B = 0$ then $k_5 + carry_C = 0 \pmod 2$, but $carry_C = C$, therefore $k_5 = C$.

If $carry_B = 1$ then $k_5 + carry_C - 2 = 0 \pmod 2$, but $carry_C = C$,

therefore, $k_5 + C - 2 = 0 \pmod 2$

If $C = 1$ then $k_5 = 1$, if $C = 0$ then $k_5 = 0$, therefore $k_5 = C$.

We conclude that: $B^* = B$ if and only if $k_5 = C$.

$$A^* = A + k_4 + carry_B - 2\,carry_A$$

..... (6)

Now $\underline{A^* <> A}$, if and only if, $k_4 + carry_B <> 0 \pmod 2$ (since $2\,carry_A$ mod 2 is 0 for $carry_A = 0$ or 1), or when $k_4 <> carry_B$.

Examine two cases $C = 0$ and $C = 1$:

When $C = 0$, then $k_5 = 0$, $carry_C = 0$, $carry_B = 0$ if and only if $k_4 = 1$,

When $C = 1$, then $k_5 = 1$, $carry_C = 1$, $carry_B = 1$ if and only if $k_4 = 0$.

We conclude, that in all cases, the key bits $k_4 <> k_5$, and $k_6 = 1$.

The input XOR, E', is limited to two values, 80 and 40 (Hex), which comply with an input difference K(E') of 28 (Hex) after key multiplication. The transition 28 (Hex) → 0, for substitution box 4, occurs with probability 16/64. This means that only 16 correct pairs (E, E*) exist which yield the output difference 0. However, each E' also has an associated 16 (E, E*) pairs (total of 32 pairs). Therefore, only half the possible (E, E*) pairs are

correct ones and this reduces the overall probability by ½ . The probability of this characteristic (as shown in Figure 20) becomes (¼) (½) = 1/8 . To be useful we must apply this characteristic iteratively over the MultiDES rounds.

Key schedule compliance

In order to use this characteristic iteratively the conditions on the key bits must be obeyed for all sixteen rounds. This must be checked via the key scheduling algorithm to ascertain that throughout the sixteen rounds no violations of these conditions are encountered. We used the key schedule of DES for our analysis, with the knowledge that even, independently generated keys should not severely alter our conclusions. The key schedule in DES involves an initial permutation which selects 56 from 64 bits, a dividing of the 56 bits into two 28 bit halves, a circular shift left 1 or 2 bits depending on round number and a permuted choice to select 48 subkey bits. At each round a different key bit assumes key bit location $k_6$ (which is really key bit location 24, counting from right to left, with respect to the round subkey) and will therefore be constrained to the value 1. Therefore 16 key bits are constrained to the value 1. In addition 16 x 2 different key bits assume key positions $k_4$, $k_5$ (locations 22 and 23,counting from right to left, with respect to the round subkey) and cannot be equal to each other. Table VI lists key bits occupying key bit locations $k_4$, $k_5$ and $k_6$ during 16 rounds.

We conclude that:

**For odd rounds key bit 18 cannot equal key bit 27.**

This is true since for round 1, 36 <> 27 and for round 5, 36 <> 9 and for

round 9,

9 <> 18, therefore **18 <> 27.**

However, we find that in round 1 key bit 18 is 1 and in round 5 key bit 27 is 1, therefore we cannot iterate very far on odd rounds using this characteristic.

**For even rounds key bit 17 cannot equal key bit 44.**

This is true since for round 8, 26 <> 17 and for round 12, 26 <> 35 and for round 16 35 <> 44, therefore **17 <> 44.**

However, we find that in round 12 key bit 17 is 1 and in round 8 key bit 44 is 1, a contradiction. Therefore, we can apply this characteristic for only **15 rounds** (incompatibility occurs in round 16, where bit 44 contradicts with bit 35 and therefore with bit 17) with 14 conditions on the key bits.

**TABLE VI**

**KEY BITS IN $k_4$, $k_5$ and $k_6$**

| ROUND NUMBER | BITS IN $k_4$, $k_5$ (<>) | BITS IN $k_6$ (=1) |
|---|---|---|
| 1 | 36 27 | 18 |
| 2 | 57 19 | 10 |
| 3 | 41 3 | 59 |
| 4 | 25 52 | 43 |
| 5 | 9 36 | 27 |
| 6 | 58 49 | 11 |
| 7 | 42 33 | 60 |
| 8 | 26 17 | 44 |

| | | |
|---|---|---|
| 9 | 18 9 | 36 |
| 10 | 2 58 | 49 |
| 11 | 51 42 | 33 |
| 12 | 35 26 | 17 |
| 13 | 19 10 | 1 |
| 14 | 3 59 | 50 |
| 15 | 43 52 | 34 |
| 16 | 44 35 | 26 |

We conclude that this 14 round characteristic has a probability of:

$$\textbf{PROB [14 rounds]}\ (28_x \rightarrow 0)_{S4} = (1/2^3)^7 = 2^{-21}$$

.....(7)

Thus the attack on this modified MultiDES requires $\sim 2^{23}$ chosen plaintexts.

Case r = 3, s = 2:

Assume the key, K, has piece entering substitution box 4 with bits $k_1$ $k_2$ $k_3$ $k_4$ $k_5$ $k_6$.

(These bits are those entering substitution box 4 and are numbered for convenience with respect to this substitution box, but they are really bits 19-24 of the subkey, as counted from left to right or bits 25 to 29 when counted from right to left.) Since

$s = 2$, $k_5 = 1$ and $k_6 = 0$.

Assume E has bits a b c d f g and E* has bits a* b* c* d* f* g* ,entering substitution box 4.

To comply with the lemma and its corollary, E'= 000004000000 (Hex), and

therefore, d must compliment d*. Without loss of generality we can write: a = a*,

b =b*, c = c*, d <> d*, f = f* and g = g*. We can also set d = 0, causing d* = 1.

Consider the expression:

$$E^* \cdot K = (E \cdot K) + (E' \cdot K)$$

.....(3)

We evaluate the right hand side of Eq. (3).

$(E' \cdot K) = (0\ 0\ 0\ 1\ 0\ 0) \cdot (k_1\ k_2\ k_3\ k_4\ 1\ 0)$ whose right half pattern is $k_3\ k_4\ 1\ 0\ 0\ 0$.

$(E \cdot K)$ is unknown and assumed to have bit pattern: A B C D F G.

$(E^* \cdot K)$ is given Eq. (3) and evaluated :

| | A | B | C | D | F | G |
|---|---|---|---|---|---|---|
| + | | | | | | |
| | $k_3$ | $k_4$ | 1 | 0 | 0 | 0 |
| | A* | B* | C* | D | F | G |

The following conditions are automatically valid: C <> C*, D = D*, F = F*, G = G*,

$carry_G$, $carry_F$, and $carry_D$, all equal zero. In addition; since, K(E') (Eq. (1)) was set at (Hex) 28, (bit pattern 1 0 1 0 0 0), A <> A*, and B = B*. We investigate the carry arithmetic for our conditions B = B*, A <> A*:

$$B^* = B + k_4 + carry_C - 2\ carry_B$$

..... (8)

Now $\underline{B^* = B}$, if and only if, $k_4 + carry_C = 0 \pmod 2$, or when $k_4 = carry_C$

. But $\mathbf{carry_C} = C$, therefore $\mathbf{k_4} = C$.

$$A^* = A + k_3 + \mathbf{carry_B} - 2\,\mathbf{carry_A}$$

..... (9)

Now $A^* <> A$, if and only if, $k_3 + \mathbf{carry_B} <> 0 \pmod 2$, or when $k_3 <> \mathbf{carry_B}$.

Examine two cases $C = 0$ and $C = 1$:

When $C = 0$, then $k_4 = 0$, $\mathbf{carry_C} = 0$, $\mathbf{carry_B} = 0$ if and only if $k_3 = 1$,

When $C = 1$, then $k_4 = 1$, $\mathbf{carry_C} = 1$, $\mathbf{carry_B} = 1$ if and only if $k_3 = 0$.

We conclude, that in all cases, the key bits $k_3 <> k_4$, $k_5 = 1$ and $k_6 = 0$.

The given constraints reduce the overall probability by ½ (as noted in discussion of previous case). The probability of the input difference to achieve the desired output difference in substitution box 4; i.e, $\text{Prob}[28_x \rightarrow 0]_{S4}$ is ¼ ; therefore, the probability of this characteristic is (1/4)(1/2) = 1/8. To be useful we must apply this characteristic iteratively over the MultiDES rounds.

We check for compliance with the key schedule. Table VII is a list of subkey bits which occupy key positions $\mathbf{k_3}$, $\mathbf{k_4}$, **and** $\mathbf{k_5}$.

We conclude that:

**For odd rounds:**

From Table VII (round 1) key bit 1 cannot equal key bit 36; however, from Table VI we find that key for round 13 key bit 1 = 1 and for round 9 key bit 36 = 1. Other contradictions exist.

**For even rounds:**

From Table VII, $k_3$ cannot equal $k_4$; however, in round 2 $k_3 = 58$ and in

round 6

$k_4 = 58$. In addition, for round 4 $k_3 = 26$ and in round 8 $k_4 = 26$.

Therefore, we cannot apply this characteristic for 16 rounds.

This characteristic is useless for a complete differential cryptanalysis attack. It may be useful for independent keys which are not constrained to the DES key schedule algorithm.

**MultiDES, general case**

We proceed to apply our methodology used on the isolated substitution box case, to the general case of MultiDES, in which common multiplication may affect neighboring substitution boxes. Again we will show **MultiDES** is **stronger** than DES in a differential cryptanalysis attack, and that the key space for compliance with high probability characteristics is limited, and the probability of success for the limited key space is less than in an attack against DES.

In order to apply the techniques used previously, we note, that only for the first substitution box can the affect of common multiplication resemble the isolated substitution box case. Multiplying two bit patterns each of length **i**, results in a bit pattern of length **2i-1**. This causes bit patterns of an input difference **E' (E XOR E*)** belonging to a higher substitution box, to affect, after key multiplication, the input difference pattern **K(E')** of a lower substitution box. Since we can choose the bit pattern of **E'** so that all bits not associated with substitution box 1 are set to zero (e.g., bits 1-42, counting from right to left of the round subkey), the desired substitution box will not be affected after key multiplication, by bit patterns of **E'**, or of those of the key, that were not associated with substitution box 1 before the

multiplication.

TABLE VII

KEY BITS IN $k_3$, $k_4$ and $k_5$

| ROUND NUMBER | BITS IN $k_3$, $k_4$ (<>) | BITS IN $k_5$ (=1) |
|---|---|---|
| 1 | 1 36 | 27 |
| 2 | 57 58 | 19 |
| 3 | 42 41 | 3 |
| 4 | 26 25 | 52 |
| 5 | 10 9 | 36 |
| 6 | 59 58 | 49 |
| 7 | 43 42 | 33 |
| 8 | 27 26 | 17 |
| 9 | 19 18 | 9 |
| 10 | 3 2 | 58 |
| 11 | 52 51 | 42 |
| 12 | 36 35 | 26 |
| 13 | 49 19 | 10 |
| 14 | 33 3 | 59 |
| 15 | 17 52 | 43 |
| 16 | 9 44 | 35 |

Moreover, for substitution box 1 we have no concern with bit interaction to a lower substitution box as a result of the multiplication step. We therefore,

select substitution box 1 for our attack. The input difference, **K(E')**, into substitution box 1 must only affect this box; hence, the last two bits of its bit pattern must be zero. Moreover, we seek an input difference which yields an iterative characteristic of the form shown in Fig.12, i.e., one yielding a transition to an output difference of zero with high probability. The highest probability input difference for substitution box 1 yielding output difference zero (obtained from the Difference Distribution Table) is 28 (Hex), with probability:

Prob (28→0) is 12/64 = 3/16

Applying the Lemma to 28 (hex) ( bit pattern 101000), in substitution box 1, gives **r +s –1 = 46**, or **r + s = 47** (counting from the right, bits 43-48 enter substitution box 1). As noted previously, **E, E* and E'** must be valid e-expansions and E' not affect previous or next substitution boxes. The nibbles

entering substitution box 1 are nibbles 1 and 2 of the input XOR, **E'** (all of nibble 1

and the leftmost 2 bits of nibble 2). To be a valid e-expansion, nibble 2 must be

either **0, 5, A,** or **F**. Only **0** for nibble 2 agrees with the condition that the input XOR

will have its two rightmost bits both zero. This limits the possibilities for an **E'** associated with substitution box 1 to **00, 10, 20, 30. E'= 00** is not useful for a differential attack .Applying the Lemma to the case at point, input difference to substitution box 1 having Hex bit pattern 28 (001010), **r + s –1**

= 46, r + s = 47. This means there are several choices for characteristics:

for r = 46, s = 1 (E' = 20, 30)

for r = 45, s = 2 (E' = 10)

Applying the Corollary of the Lemma, we eliminate E'=30 (Hex).

This constrains the choices of E' and the corresponding keys which can give a given K(E').

These r values (together with the corollary) limit the input XOR, E', into substitution box 1, to 20, and 10 (Hex).

These s values give us our first constraints on the subkey bits entering substitution box 1. Allowable keys have bit patterns of xxxxx1 or xxxx10 (where x can be arbitrarily 0 or 1). We now show additional constraints on the key bits entering substitution box 1.

We select r = 46 or E' = 20 (Hex) (with bit pattern 0 01 0 0 0 0 0), giving s = 1, and

therefore the last bit of the subkey entering substitution box 1, $k_6 = 1$.

In an analysis identical to that for the isolated substitution box case, (and repeated here, in part, for convenience) we obtain the following:

| | | | | | | |
|---|---|---|---|---|---|---|
| (E . K): | A | B | C | D | F | G |
| + | | | | | | |
| (E' . K): | $k_4$ | $k_5$ | 1 | 0 | 0 | 0 |
| | A* | B* | C* | D* | F* | G* |

Therefore C <> C*, D = D*, F = F*, and G = G*, with $carry_G$, $carry_F$, $carry_D$, all equal zero ($carry_H$, the carry into G* is also zero since bits 1-42, conting from right to left, of E' are all zero) and $carry_C$ = C.

Since our input difference **K(E')** = 28 (Hex), **A** XOR **A*** = 1, **B** XOR **B*** = 0, and **A** <> **A***, and **B** = **B***.

We examine the carry arithmetic for **B, B*** and **A, A***.

$$\mathbf{B} = \mathbf{B}^* + k_5 + \mathbf{carry}_C - 2\mathbf{carry}_B$$

.....(5)

**B** = **B*** if and only if $k_5 + \mathbf{carry}_C = 0 \pmod 2$, $-2\ \mathbf{carry}_B$ is zero mod 2 for $\mathbf{carry}_B = 0$ or 1.

Therefore, $k_5 = \mathbf{carry}_C$. But $\mathbf{carry}_C = C$, therefore $k_5 = C$.

$$\mathbf{A} = \mathbf{A}^* + k4 + \mathbf{carry}_B - 2\mathbf{carry}_A$$

..... (6)

**A** <> **A*** if and only if $k_4 + \mathbf{carry}_B \neq 0 \pmod 2$, therefore $k_4 \neq \text{carry}_B$.

Examine two cases **C** = 0 and **C** = 1:

When **C** = 0, then $k_5 = 0$, $\mathbf{carry}_C = 0$, $\mathbf{carry}_B = 0$ if and only if $k_4 = 1$,

When **C** = 1, then $k_5 = 1$, $\mathbf{carry}_C = 1$, $\mathbf{carry}_B = 1$ if and only if $k_4 = 0$.

We conclude, that in all cases, the key bits $k_4 \neq k_5$, and $k_6 = 1$.

The input XOR, **E'**, is limited to two values, 10 and 20 (Hex), which comply with an input difference **K(E')** of 28 (Hex) after key multiplication. The transition , 28 (Hex) → 0, for substitution box 1, occurs with probability 12/64. This means that only 12 correct pairs (**E**, **E***) exist which yield the output difference 0. However, each **E'** also has an associated 12 (**E**, **E***) pairs (total of 24 pairs). Therefore, only half the possible (**E**, **E***) pairs are correct ones and this reduces the overall probability by ½ .

Therefore, the probability for this iterative characteristic is:

$$\textbf{PROB [1 round]}\ (28_x \rightarrow 0)_{S1} = (1/2)*(3/2^4) = 2^{-3.42} \qquad \text{.....(10)}$$

To be useful we must apply this characteristic iteratively over the MultiDES rounds.

In order to use this characteristic iteratively we check the key schedule for compliance. We find conflicts with the key schedule for round 16. Therefore, we can only use this characteristic for 14 rounds.

We conclude that this 14 round characteristic has a probability of:

$$\textbf{PROB [14 rounds]}\ (28_x \rightarrow 0)_{S1} = (2^{-3.42})^7 = 2^{-23.94} \sim 2^{-24} \qquad \text{.....(11)}$$

In summary we note the following:

This probability is for a given best characteristic, which applies only to specific keys. Such keys are rare. In the case above, although the characteristic has probability $\sim 2^{-24}$, it only applies to $2^{-28}$ of the possible keys. Thus we conclude, their exists many keys for which there are no good characteristics by which to attack them. Therefore, MultiDES and its variants are cryptographically **stronger** than DES.

## TABLE II

### E - Expansion

| | | | | | |
|---|---|---|---|---|---|
| 32 | 1 | 2 | 3 | 4 | 5 |
| 4 | 5 | 6 | 7 | 8 | 9 |
| 8 | 9 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 30 | 31 | 32 | 1 |

## TABLE III

### Substitution Boxes

### Substitution box 1

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 14 | 4 | 13 | 1 | 2 | 15 | 11 | 8 | 3 | 10 | 6 | 12 | 5 | 9 | 0 | 7 |
| 1 | 0 | 15 | 7 | 4 | 14 | 2 | 13 | 1 | 10 | 6 | 12 | 11 | 9 | 5 | 3 | 8 |
| 2 | 4 | 1 | 14 | 8 | 13 | 6 | 2 | 11 | 15 | 12 | 9 | 7 | 3 | 10 | 5 | 0 |
| 3 | 15 | 12 | 8 | 2 | 4 | 9 | 1 | 7 | 5 | 11 | 3 | 14 | 10 | 0 | 6 | 13 |

### Substitution box 2

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 15 | 1 | 8 | 14 | 6 | 11 | 3 | 4 | 9 | 7 | 2 | 13 | 12 | 0 | 5 | 10 |
| 1 | 3 | 13 | 4 | 7 | 15 | 2 | 8 | 14 | 12 | 0 | 1 | 10 | 6 | 9 | 11 | 5 |
| 2 | 0 | 14 | 7 | 11 | 10 | 4 | 13 | 1 | 5 | 8 | 12 | 6 | 9 | 3 | 2 | 15 |
| 3 | 13 | 8 | 10 | 1 | 3 | 15 | 4 | 2 | 11 | 6 | 7 | 12 | 0 | 5 | 14 | 9 |

### Substitution box 3

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 10 | 0 | 9 | 14 | 6 | 3 | 15 | 5 | 1 | 13 | 12 | 7 | 11 | 4 | 2 | 8 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 13 | 7 | 0 | 9 | 3 | 4 | 6 | 10 | 2 | 8 | 5 | 14 | 12 | 11 | 15 | 1 |
| 2 | 13 | 6 | 4 | 9 | 8 | 15 | 3 | 0 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |
| 3 | 1 | 10 | 13 | 0 | 6 | 9 | 8 | 7 | 4 | 15 | 14 | 3 | 11 | 5 | 2 | 12 |

***Substitution box 4***

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 7 | 13 | 14 | 3 | 0 | 6 | 9 | 10 | 1 | 2 | 8 | 5 | 11 | 12 | 4 | 15 |
| 1 | 13 | 8 | 11 | 5 | 6 | 15 | 0 | 3 | 4 | 7 | 2 | 12 | 1 | 10 | 14 | 9 |
| 2 | 10 | 6 | 9 | 0 | 12 | 11 | 7 | 13 | 15 | 1 | 3 | 14 | 5 | 2 | 8 | 4 |
| 3 | 3 | 15 | 0 | 6 | 10 | 1 | 13 | 8 | 9 | 4 | 5 | 11 | 12 | 7 | 2 | 14 |

***Substitution box 5***

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2 | 12 | 4 | 1 | 7 | 10 | 11 | 6 | 8 | 5 | 3 | 15 | 13 | 0 | 14 | 9 |
| 1 | 14 | 11 | 2 | 12 | 4 | 7 | 13 | 1 | 5 | 0 | 15 | 10 | 3 | 9 | 8 | 6 |
| 2 | 4 | 2 | 1 | 11 | 10 | 13 | 7 | 8 | 15 | 9 | 12 | 5 | 6 | 3 | 0 | 14 |
| 3 | 11 | 8 | 12 | 7 | 1 | 14 | 2 | 13 | 6 | 15 | 0 | 9 | 10 | 4 | 5 | 3 |

***Substitution box 6***

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 12 | 1 | 10 | 15 | 9 | 2 | 6 | 8 | 0 | 13 | 3 | 4 | 14 | 7 | 5 | 11 |
| 1 | 10 | 15 | 4 | 2 | 7 | 12 | 9 | 5 | 6 | 1 | 13 | 14 | 0 | 11 | 3 | 8 |
| 2 | 9 | 14 | 15 | 5 | 2 | 8 | 12 | 3 | 7 | 0 | 4 | 10 | 1 | 13 | 11 | 6 |
| 3 | 4 | 3 | 2 | 12 | 9 | 5 | 15 | 10 | 11 | 14 | 1 | 7 | 6 | 0 | 8 | 13 |

***Substitution box 7***

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 4 | 11 | 2 | 14 | 15 | 0 | 8 | 13 | 3 | 12 | 9 | 7 | 5 | 10 | 6 | 1 |
| 1 | 13 | 0 | 11 | 7 | 4 | 9 | 1 | 10 | 14 | 3 | 5 | 12 | 2 | 15 | 8 | 6 |
| 2 | 1 | 4 | 11 | 13 | 12 | 3 | 7 | 14 | 10 | 15 | 6 | 8 | 0 | 5 | 9 | 2 |
| 3 | 6 | 11 | 13 | 8 | 1 | 4 | 10 | 7 | 9 | 5 | 0 | 15 | 14 | 2 | 3 | 12 |

***Substitution box 8***

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 13 | 2 | 8 | 4 | 6 | 15 | 11 | 1 | 10 | 9 | 3 | 14 | 5 | 0 | 12 | 7 |
| 1 | 1 | 15 | 13 | 8 | 10 | 3 | 7 | 4 | 12 | 5 | 6 | 11 | 0 | 14 | 9 | 2 |
| 2 | 7 | 11 | 4 | 1 | 9 | 12 | 14 | 2 | 0 | 6 | 10 | 13 | 15 | 3 | 5 | 8 |
| 3 | 2 | 1 | 14 | 7 | 4 | 10 | 8 | 13 | 15 | 12 | 9 | 0 | 3 | 5 | 6 | 11 |

*TABLE IV*

***P - Permutation***

| | | | |
|---|---|---|---|
| 16 | 7 | 20 | 21 |
| 29 | 12 | 28 | 17 |
| 1 | 15 | 23 | 26 |
| 5 | 18 | 31 | 10 |
| 2 | 8 | 24 | 14 |
| 32 | 27 | 3 | 9 |
| 19 | 13 | 30 | 6 |
| 22 | 11 | 4 | 25 |

*TABLE V*

*S-BOXES from DES prior art, in different format*

*static char si[8][64] = {14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7, 0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8, 4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0, 15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13, 15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10, 3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5, 10, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15, 13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9, 10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8, 13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1, 13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7, 1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12, 7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15, 13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9, 10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4, 3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14, 2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9, 14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6, 4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14, 11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3, 12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11, 10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8, 9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6, 4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13, 4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1, 13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6, 1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2, 6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12, 13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7, 1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2, 7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8, 2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11};*

## APPENDICES: Add - DES (detailed analysis)

In DES there are two XOR - operations in each round. The first XORs the expanded input with the subkey within the F function while the other XORs, the output of the F function with the other half of the input data. The following describes the possible modification of DES which replace the XOR with the F function by addition operation. This variant of cryptosistem we call "add - DES."
We are going to apply the technique of differential cryptanalysis to add - DES.
First of all some preliminary remarks mainly of heuristic character.

1. According to general scheme of Biham and Shamir [BiSh93] we must choose the proper characteristic as a base for iterative characteristic. The term "proper" means that its probability must be better then probability of known one for original version of DES / * We want to compare the cryptographic strength of both systems with initial conjecture that add - DES is weaker than DES * /
2. Simple observation shows that the key addition of key K to each member of plain texts pair (E,E*) in general does change the XOR result E' = E+E* of this pair:
   $K(E)' = (K+E) \oplus (K+E^*) \neq E+E^* = E'$
3. Therefore one must describe the necessary conditions on right (P,P*) pair and maybe on "weak keys" / * a priori, we do not know whether we can develop a successful attach on any key, or only on special "weak" one * / which imply possibility of evaluation XOR result K(E)' after key addition.
4. Such possibility permits us to choose efficiently (P,P*) pair with prescribed XOR K(E)' for which one can apply different distributive tables DDT with best possible probability.
5. However what was said in paragraph 2, there is a class of (P,P*) pairs, for which addition conserves the XOR result, we mean (E,E*) pair with E'=0 , i.e. E=E*. Therefore when the XOR of the pairs are zero the outputs are equal too which makes all the keys equally likely.
6. The above point implies that we must try (E,E*) pair with E'≠0. In this case we obtain the next invariant: Key addition conserves the position of the last non zero bit in E' - XOR.
7. If this last non zero bit in E' - XOR belongs to $S_i$ - box then we can say something more definite about its position in S: If $i \neq 8$ then the last two bits of XOR - input for $S_i$ - are zero.
   Indeed otherwise according to definition of E - expansion , the last two bits in $S_i$ box coincide exactly with leading two bits of $S_{i+1}$ box. This contradicts with the choice of $S_i$ box.
8. One can prove such statement: If $S_i$ - the only S box, for which E' differ from zero, E' ≠ 0, than for any key K the probability of $(K(E)' \rightarrow 0) = 0$.
9. The proof of above statement shows that the result XOR of key - addition K(E') =(K+E)⊕(K+E*) in this last important $S_i$ - box can be only one of the next values /* in hex signature */ $N_x$ = 4, 8, C, 10, $14_x$, $18_x$, $1C_x$, $20_x$, $24_x$, $28_x$, $2C_x$, $30_x$, $34_x$, $38_x$, $3C_x$. Therefore one can choose the best candidates for an attack: the best important $S_i$ box together with proper $N_x$ . For example:
   $S_i = S_4$, and $N_x = 28_x = 101000$.
10. Some remarks about carry. We are going to calculate all carry in both additions: K+E and K+E*. Suppose we have sum U+V=W in binary signature with corresponding bits $u_i$ , $v_i$ and $w_i$ in i-th column, i = 0, 1, ..., n. The preliminary value of sum in i+1 - th column is $Sum_{i+1} = u_{i+1} + v_{i+1} + carry_i$. The real value of sum in i+1 - th column is $w_i+1 = Sum_{i+1} \pmod 2$. Since $u_0+v_0 <= 1+1=2$ we have $carry_0 <= 1$. Suppose that $0 <= carry_i <= 1$. Then $Sum_{i+1} = u_{i+1}+v_{i+1}+ carry_i <= 3$. Therefore $0 <= carry_{i+1} = (Sum_{i+1}-w_{i+1}) / 2 <= 1$. So always $0 <= carry_i <= 1$, $w_i = Sum_i - 2\, carry_i$.
11. Now we can research an attack in [BiSh93]: The last important S - box is $S_4$, $N_x = 28_x$ with $p(N_x \rightarrow 0) = 16/64$ (the best possible probability for an isolated S - box. The pair ( P,P* ) of plaintext with XOR I' of right halves. I' = 000e0000 = 0000 which after E - expansion in:

0000
0000
1100
0000
0000
0000
0000

the first round transfers into the E' = 000000 = 000058000000.
000000

000001
011000
000000
000000
000000
000000

The authors [BiSh93] declare that the addition of the key K cause the input XOR to become
$K(E)' = (K+E)\oplus(K+E^*) = 000028000000 = 000000$

000000
000000
101000
000000
000000
000000
000000

with probability of 1/16. Let us consider an arbitrary ( E,E* ) pair with fixed gives E' - XOR.
According to choice we have for S3 and $S_4$ boxes:

| | $S_3$ | $S_4$ | $S_5$ | $S_6$ | |
|---|---|---|---|---|---|
| E = | . . . . a | b x | y z | t . . | . . . . a b` | x` y` z` t . . . = E* |
| + | | | | | + |
| K = | . . . . A | B X | Y Z | T . . | . . . . A B | X Y Z T . . . = K |
| K+E = | . . . . A | B X | Y Z | T . . | . . . . A* B* | X* Y* Z* T* . . . = K+E |

( Here the symbols b and b` denote opposite values. )
We denote corresponding carry in a-, b-, x-, y-, ... position for K + E and K + E* by $carry_a$,$carry_b$, and by $carry_a^*$, $carry_b^*$, ... respectively.
Let us make some observations about such carries.

12. From the above ( paragraph number 10):
$A = a + A + carry_b - 2carry_a$ ; $A^* = a + A + carry_b^* - 2carry_a^*$.
According to assumption about key addition: $A^* = A$ hence $carry_b - 2carry_a = carry_b^* - 2carry_a^*$.
Suppose that $carry_b \neq carry_b^*$. Then one of the parts of both equations has value, belonging to {0, -2} while the other belongs to {1, -1}. Since $\{0, -2\} \cap \{1, -1\} = \emptyset$ this is contradiction.
Therefore $carry_b = carry_b^*$ and as a consequence $carry_a = carry_a^*$ .
13. Without loss of generality one can assume that b = 0, b` = 1. In other case we can swap 1 and 1*.
$B = b + B + carry_a - 2carry_b$ ; $B^* = b` + B + carry_a^* - 2carry_b^*$.
Since $B = B^*$ we have $carry_a = 1 + carry_a^*$. Therefore $carry_a = 1$ and $carry_a^* = 0$ .
14. $X = x + X + carry_y - 2carry_x$ ; $X^* = x + X + carry_y^* - 2carry_x^*$.
According to assumption $X^* = X`$, so $-(x+X) = carry_y - 2 - X$ ; $-(x+X) = carry_y^* - X`$
hence $carry_y - 2 - X = carry_y^* - X`$. Therefore:
$carry_y = carry_y^* + 2 + (X - X`) >= carry_y^* + 2 + (0-1) = carry_y^* + 1 > carry_y^*$.
$X >= 0$ ; $X` <= 1$
Thus $carry_y = 1$ and $carry_y^* = 0$
15. $Y = y + Y + carry_z - 2carry_y$ ; $Y^* = y` + Y + carry_z^* - 2carry_y^*$
According to assumption $Y^* = Y$, so $y + carry_z - 2 = y` + carry_y^*$ or $y - y` - 2 = carry_z^* - carry_z$ .
However $y - y` - 2 = -1$ . if $y = 1$ while $carry_z^* - carry_z = -1$ . if $carry_z^* < carry_z$
$-3$ . if $y = 0$ $0$ . if $carry_z^* = carry_z$
$1$ . if $carry_z^* > carry_z$
Therefore for the only common value of both parts, equal 1, we have $y = 1$ and $y` = 0$ .
16. We see that b = 0, y = 1 and b` = 1, y` = 0. The value of b- and y- bits in E (likely as value of corresponding bits in E* ) are opposite to each other. However from the definition of E - expansion these value must be coincide. This is contradiction. So, instead of statement the authors:
There does not exist such a key K, that its addition cause the input XOR E' = 000058000000 to become K(E)' = 000028000000.

WHAT IS CLAIMED:

(1) A method for operating a general purpose data processor to enable said data processor to encrypt, the method comprising the steps of:

(a) employing an arithmetic operation on a plurality of single size inputs yielding said plurality size single result, and

(b) folding a distinct single size portion of said result in said plurality companion executions.

(2) A machine for encrypting plain text-derived-input comprising:

(a) a memory providing the s-boxes of DES as numbers, and

(b) a combiner combining said numbers on a bit-by-bit basis with limited carry into the stream of said plain text-derived-input.

(3) A method for a cryptographic primative to enable a data processor to perform said cryptographic primative on plain-text derived input, the method comprising the steps of:

(a) selecting a cryptographically suitable mask depending on information available within the round function selected from the group of round number. block number, and data being encrypted, and

(b) combining said mask into stream of said plain text-derived-input.

(4) A cryptobox machine comprising:

(a) a plain text provider providing a plain text,

(b) a subkey provider providing a plurality of subkeys,

(c) a cryptobox employing said plain text and said subkeys to generate new subkeys, and

(d) a connector providing said new subkeys to another cryptobox to process more plain texts.

(5) A method for implementing substitution boxes in logic gates on a 32-bit microprocessor, the method comprising the steps of:

(a) calculating combinations of 32-bit variables for repeated usage, and

(b) employing 32-bit equations which accurately calculate a value of a single bit of output of the substitution box using the results of step (a).

(6) A machine for data scrambling comprising a local scrambling operation and a permutation distributing bits from output of a given local scrambler to input of other local scramblers, comprising:

(a) a local scrambler P which distributes four outputs among eight possible boxes, and

(b) a global scrambler PP which distributes a plurality of outputs among groups of possible s-boxes to effect an extended P permutation.

(7) A DES encryption method comprising: performing N DES rounds, including, for at least one 1<-n<=N, performing an n'th DES round on a subkey and a plain text derived input to said n'th round wherein addition is substituted for exclusive-or in performing said n'th DES round, wherein a subkey is defined for each of said N rounds.

(8) A WDES encryption method comprising: performing a plurality of rounds of WDES encryption, each round using a round function F; wherein, for the round function F of at least one round, a form of multiplication is substituted for exclusive-or.

(9) A method for performing a round function of an iterated encryption for a plurality of 32-bit input blocks, the steps of the method being performed by a data processor, the method comprising the steps of:

(a) numbering the plurality of input blocks from "0" to "n" with an input block number;

(b) splitting each of the plurality of input blocks into an upper half and a lower half to produce plain text-derived input;

(c) combining said plain text-derived input with a plurality of round-dependent subkeys according to a form of multiplication to form a blended product;

(d) applying a plurality of s-boxes of the F function of a DES encryption algorithm to said blended product; and

(e) applying the P permutation of the F function of a DES encryption algorithm to output of step (d).

(10) The method of claim 9, further comprising:

(a) applying said plurality of s-boxes in bit-slice form using logic gates.

(11) The method of claim 9, further comprising:

(a) selecting a mask determined according to a criteria selected from the group of a number of a round being performed and said input block number, and

(b) combining said mask with said plain text-derived input.

(12) The method of claim 9, wherein said form of multiplication features the steps of:

(I) multiplying a plurality of bits from said plain text-derived input and a plurality of bits from said plurality of round-dependent subkeys to form a common multiplication product; and

(II) performing an exclusive-or function on a plurality of bits from said plain text-derived input and a plurality of bits from said plurality of round-dependent subkeys to form a balanced product.

(13) The method of claim 12, wherein the step of combining said plain text-derived input with a plurality of round-dependent subkeys further comprises the steps of:

(III) performing an addition function on said common multiplication product and said balanced product to form a pseudo-random product.

(14) The method of claim 13, wherein the step of combining said plain text-derived input with a plurality of round-dependent subkeys further comprises the steps of performing a thorough folding operation on two pseudo-random products as follows:

(IV) folding upper half of first pseudo random product into lower half of second pseudo random product to form first result,

(V) folding lower half of first pseudo random product into upper half of second pseudo random product to form second result, and

(VI) concatenating first result to second result to form folded product.

(15) The method of claim 14, wherein the step of combining said plain text-derived input with a plurality of round-dependent subkeys further comprises the steps of performing a blending operation on two folded products as follows:

(VII) concatenating lower half of first folded product with upper half of second folded product to form said blended product.

(16) The method of claim 14, wherein said folding operation is exclusive-or.

(17) A machine for performing a cryptographic primative comprising:

(a) a key-inserter which employs a form of multiplication for key insertion, whereby block length of the cryptographic primative is extended.

(18) A machine according to claim 17, wherein said form of multiplication in said key inserter comprises:

(a) a multiplier which performs an operation as follows (a*b)'(a exclusive-or b).

(19) A machine according to claim 17, further comprising:

(a) an associator embodying a look-up-table implemented by bit-slicing.

(20) A machine according to claim 17, further comprising:

(a) a multiplier which performs an operation in chunks as least as large as a byte.

(21) A machine according to claim 17, further comprising:

(a) a multiplier which performs individual multiplications over a Fermat filed with only a less than a logarithmic, in size of the field, number of exceptions.

(22) A machine according to claim 17, wherein said form of multiplication in said key inserter comprises:

(a) a multiplier, performing common multiplication of arguments to yield a product,

(b) a designator, designating the upper and lower half of said product,

(c) a combiner, combining the upper half with the lower half using exclusive-or to form a final product, whereby the final product maintains behavior of modulo multiplication without the clear algebraic structure.

(23) A machine according to claim 17, wherein said form of multiplication in said key inserter comprises:

(a) a first multiplier, performing common multiplication of arguments to yield a first product,

(b) a second multiplier, performing common multiplication of other arguments to yield a second product,

(c) a first designator, designating a upper and lower half of said first product,

(d) a second designator, designating a upper and lower half of said second product,

(e) a first combiner, combining the upper half of the first product with the lower half of the second product using exclusive-or to form a first final product,

(f) a second combiner, combining the upper half of the second product with the lower half of the first product using exclusive-or to form a second final product, whereby enabling folding the result of the form of multiplication with a companion execution.

(24) A machine according to claim 17, wherein said form of multiplication in said key inserter comprises:

(a) a multiplier to perform multiplication on a plurality of arguments to form a first product;

(b) a first combiner to perform exclusive or on said plurality of arguments to form a second product;

(c) a second combiner to perform addition between said first product and said second product to form a gorilla product.

(25) A folding machine according to claim 24 wherein said gorilla product is provided to a machine comprising:

(a) a counter which counts said plurality of arguments, calling it *n*;

(a) a repeater which provides a new set of arguments and calculates *n* gorilla products;

(b) a splitter which divides each gorilla product into *n* pieces, each with index *i* from *1..n*;

(c) a combiner which combines using exclusive-or *n* pieces such that the combine will take exactly one piece from each gorilla product, and exactly one piece of any gorilla product with the index *i* for all 1, such that said combiner yields a plurality of *n* folded products.

(26) A method for constructing a key schedule for an encryption algorithm, the steps of the method being performed by a data processor, the method comprising the steps of:

(a) determining a first set of at least one subkey for the encryption algorithm;

(b) encrypting a master key according to the encryption algorithm by using said first set of at least one subkey to product a cipher text;

(c) repeating step (b) for at least a first number of rounds required to achieve dependence of every bit of said cipher text on each bit of said master key;

(d) repeating step (b) for an integral number of rounds, said integral number being at least one, extracting subkeys from output of said round(s).

(e) repeating step (d) until a second set of subkeys has been generated.

(27) The method of claim 26, further comprising the steps of:

(i) deriving said first set of at least one subkey from DES s-box entries.

(28) The method of claim 26, further comprising the steps of:

(i) deriving said second set of at least one subkey from the group of the output and intermediate values of round function in the encryption algorithm.

(29) The method of claim 26, further comprising the steps of:

(f) encrypting said cipher text with said second set of at least one subkey according to the encryption algorithm to produce further encrypted cipher text, such that a third set of subkeys is created for use in encryption of actual plain text.

(30) A symmetric cryptobox machine comprising:

(a) circuits employing at least a 128-bit key and block size.

(31) A machine according to claim 30, wherein

(a) said circuits providing large key size are implemented by employing circuits providing a large block size.

(32) A machine according to claim 30, further comprising:

(a) an optimal sorting network performing combining.

(33) A method for operating a general purpose data processor of known type to enable said data processor to encrypt employing a key schedule the method comprising the steps of:

(a) feeding the full set of 64 key bits per block into a rearranged PC2 from DES.

(34) A method according to claim 33, further comprising:

(i) adding four to entries of PC2 with values above 28, prior to first usage in claim 33.

(35) A method according to claim 33, further comprising:

(i) performing key schedule rotation 64 bits at once rather than two groups of 32 bits.

(36) A method according to claim 33, further comprising:

(i) causing sub key to depend on the serial number of the parallel execution.

(37) A method according to claim 33, further comprising:

(i) deriving sub key by finding a multiplicative inverse over a field.

(38) A method according to claim 33, further comprising:

(i) replacing zero sub key by a round dependent mask value.

(39) A method for automatically protecting confidentiality of information stored on a persistent storage medium, the information being organized into a plurality of files, the steps of the method comprising:

(a) protecting a plurality of files of an automatic file-by-file basis, such that each of said plurality of files is automatically protected individually according to the steps of:

(i) using a cryptosystem to encrypt said at least one file, thereby generating an encrypted file; and

(ii) storing said encrypted file on the persistent storage medium.

(40) A method for protecting confidentiality of information written on a hard disk, the method comprising:

(a) encrypting a file having a selectably known file key according to a first symmetric cryptosystem; and

(b) encrypting said selectably known file key using a second symmetric cryptosystem and a selectably known master key derived from a selectably known pass phrase using a third symmetric cryptosystem wherein said third symmetric cryptosystem is operative as a cryptographically strong hash function.

(41) The method of claim 40, wherein said first and said second symmetric cryptosystems are identical.

(42) The method of claim 41, wherein said first, said second symmetric and said third symmetric cryptosystems are performed on a plurality of 32-bit input blocks, according to the steps of:

(i) numbering the plurality of input blocks from "0" to "n" with an input block number;

(ii) splitting each of the plurality of input blocks into an upper half and a lower half to produce plain text-derived input;

(iii) combining said plain text-derived input with a plurality of round-dependent subkeys according to a form of multiplication to form a blended product;

(iv) applying the P permutation of s-boxes of the F function of a DES encryption algorithm to said blended product; and

(v) applying the P permutation of the F function of a DES encryption algorithm to output of step (iv).

(43) The method of claim 40, wherein said first and said second symmetric cryptosystems are substantially different.

(44) The method claim 40, wherein at least one of said first, said second and said third symmetric cryptosystems are performed on a plurality of 32-bit input blocks, according to the steps of:

(i) numbering the plurality of input blocks from "O" to "n" with an impute block number;

(ii) splitting each of the plurality of input blocks into an upper half and a lower half to produce plain text-derived input;

(iii) combining said plain text-derived input with a plurality of round-dependent subkeys according to a form of multiplication to form a blended product;

(iv) applying a plurality of s-boxed of the F function of a DES encryption algorithm to said blended product; and

(v) applying the P permutation of the F function of a DES encryption algorithm to output of step (iv).

(45) A method according to claim 40 further comprising the steps of:

(c) decrypting said selectably known file key using said second symmetric cryptosystem and said selectably known masterkey; and

(d) decrypting said file using said selectably known file key and said first symmetric cryptosystem.

(46) A method according to claim 40, wherein said cryptographically strong hash function comprises a MAC (message authentication code).

(47) The method according to claim 40, wherein the persistent storage medium is operated by a computational device having a sleep mode, said computational device having a RAM (random access memory), such that all information on said RAM is encrypted and written to the persistent storage medium as a single unit.

(48) The method according to claim 40, wherein the persistent storage medium is operated by a computational device having a stand-by mode, said computational device having a RAM (random access memory), such that at least a portion of information on said RAM is encrypted.

(49) A system for protecting confidentiality of information stored on a persistent storage medium, the system comprising:

(a) an automatic file-by-file information protector operative to protect a plurality of files on an automatic file-by-file basis, the information protector including :

(i) a symmetric encryptor using a symmetric cryptosystem to encrypt each of said plurality of files as an individual file, thereby to generate an encrypted individual file; and

(ii) a storage manager for the persistent storage medium operative to store the encrypted individual file on the persistent storage medium.

(50) A DES encryption method comprising:

(a) performing N DES rounds, including.

for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to said n'th round wherein addition is substituted for exclusive-or in performing said n'th DES round,

wherein a sub key is defined for each of said N rounds and wherein at least some of said N sub keys are dependent.

(51) A method according to claim 50, wherein all of said N sub keys are derived from a standard key schedule.

(52) A method according to claim 50 wherein said plain text derived input to said n'th round (n>1) comprises an output of a round previous to said n'th round.

(53) A method according to claim 50 wherein said plain text derived input to said first round comprises at least a portion of said plain text.

(54) A method according to claim 50 wherein said step of performing N DES rounds comprises performing a bit-slice implementation of DES.

(55) A method according to one of claims 9 or 50, wherein for at least one 1<-n<-N, said step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key, is performed substantially before the (n+1)th round is performed.

(56) A method according to claim 55, wherein for at least one 1<=n<=N, said step of combining a plurality of key-to--sub key operations thereby to obtain an (n | 1)th sub key is performed before the n'th round is performed.

(57) A method according to claim 55, wherein for at least one 1<=n<=N, said step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key is performed before the use of the n'th sub key.

(58) A method according to claim 55, wherein for at least one 1<=n<=N, said step of combining a plurality of key-to-sub key operations thereby to obtain an (n+1)th sub key is performed before completing the use of the n'th sub key.

(59) A DES encryption method, the steps of the method being performed by a data processor, the steps of the method comprising:

(a) performing N>16 DES rounds, including, for at least one 1< - n < - N, performing an n'th DES round a sub key and a plain text derived input to said n'th round wherein addition is substituted for exclusive-or in performing said n'th DES round.

(60) A DES encryption system comprising:

(a) an addition-based DES encryptor operative to perform N DES rounds including, for at least one 1< = n <=N,

(b) a round-performer performing an n'th DES round on a sub key and

(c) a plain-text-derived-input provider providing a plain text derived input to said n'th round wherein an adder operative to perform addition rather than exclusive-or is used to perform said n'th DES round, wherein a sub key is defined for each of said N rounds and wherein at least some of said N sub keys are dependent.

(61) A DES encryption method, the steps being performed by a date processor having 32 bit registers, the steps of the method comprising:

(a) performing N DES rounds, including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain-text derived input to said n'th round to perform a bit-slice implentation of DES.

(62) A DES encryption method, the steps being performed by a data processor having registers of fewer than 64 bits, the steps of the method comprising:

(a) performing N DES rounds, including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to said n'th round to perform bit-slice implementation of DES.

(63) A DES encryption method, the steps being performed by a data processor, the steps of the method comprising:

(a) computing a sub key for each of N DES rounds, at least some of said N sub keys being dependent, by combining a plurality of key to sub key operations into a single key to sub key operation on a DES key, thereby to provide a sub key; and

(b) performing N DES rounds.

(64) A DES encryption method, the steps being performed by a data processor, the steps of the method comprising:

(a) using first and second permutations and a mapping to perform each of N DES rounds, wherein the first permutation includes a left half of L* and a right half R* and wherein L* comprises a composition of an inverse P permutation and a left half, L, of an initial permutation, and wherein R*, comprises a composition of the inverse P permutation and a right half, R, of the initial permutation, wherein the second permutation includes a left half of L** and a right half R** and wherein L** comprises a composition of the P permutation and a left half of the final permutation, and R** comprises a composition of the P permutation and a right half of the final permutation, and, wherein the mapping comprises a composition of the P permutation with an E expansion.

(65) A DES encryption method, the steps being performed by a data processor, the steps of the method comprising:

(a) performing N DES rounds, including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to said n'th round wherein addition is substituted for exclusive-or in performing said n'th DES round, wherein said step of performing N DES rounds comprises performing a bit-slice implementation of DES.

(66) A DES encryption method, the steps being performed by a data processor, the steps of the method comprising:

(a) performing N DES rounds, including, for at least one 1<=n<=N, generating an n'th k-bit s-box input by performing an n'th DES round on a k-bit sub key and a k-bit plain text derived input to said n'th round wherein multiplication in which any carry beyond k bits is discarded, is substituted for exclusive-or in performing said n'th DES round.

(67) A method according to claim 66, wherein all of said N sub keys are derived from a standard key schedule.

(68) A method according to claim 66, wherein said plain-text derived input to said n'th round (n>1) comprises an output of a round previous to said n'th round.

(69) A method according to claim 66, wherein said plain text derived input to said first round comprises at least a portion of said plain text.

(70) A method according to claim 66, wherein N>16.

(71) A DES encryption system comprising:

(a) a DES encryptor operative to perform N>16 DES rounds, including, for at least one 1<=n<=N,

(b) a round-performer performing an n'th DES round on a sub key and

(c) a plain text derived input provider operative to provide a plain text derived input to said n'th round wherein addition is substituted for exclusive-or in performing said n'th DES round.

(72) A method according to claim 66 wherein said step of performing an n'th DES round comprises performing a bit-slice DES round.

(73) A method according to claim 66, wherein a sub key is defined for each of said N rounds and wherein at least some of said N sub keys are dependent.

(74) A DES encryption method, the steps being performed by a data processor, the steps of the method comprising:

(a) performing N DES rounds, including, for at least one 1<=n<=N, performing an n'th DES round on a sub key and a plain text derived input to said n'th round wherein addition is substituted for exclusive-or in performing said n'th DES round, wherein said step of performing N DES rounds comprises performing a bit-slice implementation of DES.

(75) A WDES encryption method, the steps being performed by a data processor, the steps of the method comprising:

(a) performing a plurality of rounds of WDES encryption each round using a round function F of at least one round, addition, with final carry neglected is substituted for exclusive or.

(76) A DES encryption method comprising: performing N DES rounds, including for at least one 1<=n<=N, generating an n'th k-bit s-box input by performing an n'th DES round on a k-bit sub key and a k-bit plain text derived input to said n'th round wherein multiplication, performed over a ring, is substituted for exclusive-or in performing said n'th DES round.

(77) A method according to claim 76, wherein said multiplication over a ring comprises multiplication over a finite field.

(78) A method according to claim 76, wherein said ring has a modulus and said modulus is a product of less than 5 primes.

(79) A method according to claim 77, wherein said ring has a modulus and said modulus is a product of less than 4 primes.

(80) A method according to claim 77, wherein said ring has a modulus and said modulus is a product of 2 primes.

(81) A method according to claim 77, wherein said ring has a modulus and said modulus is prime.

(82) A method according to claim 77, wherein said ring has a modulus and said modulus comprises a product of a plurality of primes at least one of which slightly exceeds an exponent of 256.

(83) A method according to claim 77, wherein said ring has a modulus and said modulus comprises a product of a plurality of primes at least one of which slightly exceeds an exponent of 65536 such as 65536 or $2^{32}$ or $2^{48}$ or $2^{64}$.

(84) A method according to claim77, wherein said ring has a modulus and said modulus comprises a product of a plurality of primes at least one of which is slightly less than an exponent of 256.

(85) A method according to claim 77, wherein said ring has a modulus and said modulus comprises a product of a plurality of primes at least one of which slightly less than an exponent of 65536 such as 65536 or $2^{32}$ or $2^{48}$ or $2^{64}$.

(86) A method according to claim 77, wherein said step of performing an n'th DES round comprises performing a bit-slice DES round.

(87) A DES encryption system comprising:

(a) a DES encryptor for performing N>16 DES rounds, including, for at least one 1<=n<=N, an addition-based DES engine operative to perform an n'th DES round on a sub key and a plain text derived input to said n'th round wherein addition rather than exclusive or is used to perform said n'th DES round.

(88) A DES encryption system comprising:

(a) a DES encryptor for performing N DES rounds, including, for at least one 1<=n<=N, a DES engine operative to perform an n'th DES round on a sub key and a plain text derived input to said n'th round; and

(b) a computer having registers whose size is less than 64 bits, wherein said DES encryptor is configured to perform said N DES round including performing a bit-slice implementation of DES while running on said computer.

(89) A DES encryption system comprising:

(a) a sub key computation engine operative to compute a sub key for each of N DES rounds, at least some of said N sub keys being dependent, the sub key computation engine including a single key-to-sub key operation and performing said single key-to-sub key operation on a DES key, thereby to provide a sub key; and

(b) a DES engine operative to perform N DES rounds using said N sub keys.

(90) A DES encryption system comprising:

(a) a DES encryptor using first and second permutations and a mapping to perform each of N DES rounds, the DES encryptor comprising:

(i) a first permutation provider providing the first permutation which includes a left half L* and a right half R* and wherein L* comprises a composition of an inverse P permutation and a left half L of an initial permutation, and wherein R* comprises a composition of an inverse P permutation and a right half R of an initial permutation,

(ii) a second permutation provider providing the first permutation which includes a left half L** and a right half R** wherein L** comprises a composition of the P permutation and a left half L of a final permutation, and wherein R** comprises a composition of the P permutation and a right half R of a final permutation, and

(iii) a mapping provider providing the mapping which comprises a composition of the P permutation and the E expansion.

(91) A DES encryption system comprising:

(a) a DES encryptor operative to perform N DES rounds, including an addition-based DES engine performing, for at least one 1<n<N, an n'th DES round on a sub key and a plain text derived input to said n'th round wherein addition rather than exclusive or is used in performing said n'th DES round, wherein said N DES rounds are performed by performing a bit-slice implementation of DES.

(92) A DES encryption system comprising:

(a) a DES encryptor operative to perform N DES rounds, including an s-box input provider operative to provide for at least one 1<=n<=N an n'th k-bit s-box input by performing an n'th DES round on an k-bit sub key and a k-bit plain text derived input to said n'th round wherein multiplication with any carry beyond k bits is discarded, is used, rather than using exclusive or in performing said n'th DES round.

(93) A DES encryption system comprising:

(a) a DES encryptor operative to perform N DES rounds, including an addition-based DES engine operative, for at least one 1<=n<=N, to perform an n'th DES round on a sub key and a plain text-derived-input to said n'th round wherein addition rather than exclusive or is used in performing a bit-slice implementation of DES.

(94) A WDES encryption system comprising:

(a) a WDES encryptor operative to perform a plurality of rounds of WDES encryption, each round using a round function F, said WDES encryptor including an addition-based WDES engine operative for the round function F of at least one round to perform addition with final carry neglected rather than performing exclusive or.

(95) A WDES encryption system comprising:

(a) a WDES encryptor operative to perform a plurality of rounds of WDES encryption, each round using a round function F, said WDES encryptor including a common multiplication-based WDES engine operative for the round function F of at least one round to perform common multiplication with final carry neglected rather than performing exclusive-or.

(96) A DES encryption system comprising:

(a) a DES encryptor operative to perform N DES rounds, the DES encryptor including, for at least one 1<=n<=N, an s-box input provider operative to provide an n'th k-bit s-box input by performing an n'th DES round on a k-bit sub key and a k-bit plain text derived input to said n'th round wherein said n'th DES round includes performing multiplication over a ring rather than performing exclusive-or.

(97) A method for performing a cryptographic primative employing a key schedule comprising the steps of:

(a) feeding the full set of 64 key bits per block into a rearranged PC2 from DES.

(98) A method according to claim 97, further comprising at least one of the following:

(i) adding four to entries of PC2 with values above 28, prior to first usage in claim 97.

(ii) performing key schedule rotation 64 bits at once rather than two groups of 32 bits.

(iii) causing sub key to depend on the serial number of the parallel execution.

(iv) deriving sub key by finding multiplicative inverse over a field.

(v) replacing zero sub key by a round dependent mask value.

(99) A machine for performing a cryptographic primative comprising:

(a) a key-inserter which employs a form of multiplication for key insertion, whereby block length of the cryptographic primative is extended.

(100) A machine according to claim 99, wherein

(b) a logic-gate implementation of a bit-slice representation of at least one component of said cryptographic primative.

(101) A machine according to claim 100, wherein said component is a plurality of s-boxes.

(102) A machine according to claim 99, wherein said cryptographic primative is

(b) a WDES encryptor operative to perform a plurality of rounds of WDES encryption employing a plurality of WDES-round encryptors.

(c) at least one WDES-round encryptor has a form of multiplication substituted for exclusive-or as key-inserter.

(103) A machine according to claim 99 operative on plain-text derived input, further comprising:

(b) a memory providing a series of numbers having no known concise description, and

(c) a combiner combining said numbers on a bit-by-bit basis with limited carry into the stream of said plain-text derived input.

(104) A machine according to claim 103, wherein said memory provides the s-boxes of DES as numbers.

(105) A machine according to claim 103, wherein said memory provides digits selected from the group consisting of mathematical constants pi and e.

(106) A machine according to claim 99, operative to extend the effect of the P permutation comprising:

(b) a permuter whose local effect, within a group of 8 s-boxes, is identical to that of said P permutation, and

(c) said permuter whose global effect is to apply a reduced P permutation between a collection of s-boxes.

(107) A machine according to claim 99 on a plurality of input blocks, wherein

(b) a splitter operative to split each of the plurality of inputs blocks into an upper half and a lower half to produce plain text derived input.

(c) a combiner combining said plaintext derived input with a plurality of round dependent subkeys according to a form of multiplication to form a blended product in at least one round:

(d) a sbox engine applying a plurality of s-boxes of the F function of DES encryption algorithm to said blended product; and

(e) a perimeter applying the P permutation of output of said engine.

(108) A machine according to claim 107, wherein

(c) said combiner combines said plaintext derived input with a plurality of round dependent subkeys by employing addition as the form of multiplication to form a blended product.

(109) A machine according to claim 99, wherein said form of multiplication in said key inserter comprises:

(a) a multiplier performing common multiplication of arguments to yield a product.

(b) a designator, designating an upper and a lower half of said product.

(c) a combiner, combining the upper half with the lower half employing exclusive-or to form a final product.

(110) A machine according to claim 99, wherein said form of multiplication in said key inserter comprises.

(a) a first multiplier, performing common multiplication of arguments to yield a first product.

(b) a second multiplier, performing common multiplication of other arguments to yield a second product.

(c) a first designator, designating an upper and lower half of said first product.

(d) a second designator, designating an upper and lower half of said second product.

(e) a first combiner, combining the upper half of the first product with the lower half of the second product using exclusive-or to form a first final product.

(f) a second combiner, combining the upper half of the second product with the lower half of the first product using exclusive-or to form a second final product.

(111) A machine according to claim 99, wherein said form of multiplication operative in the cryptographic primative with plaintext derived input further comprises:

(i) a multiplier operative to multiply a plurality of bits from said plaintext derived input.

(112) A machine according to claim 99, operative as a hash function.

(a) said hash function providing ciphertext output.

(b) a folder providing a folding operation on said output.

(113) A machine according to claim 99, comprising:

(a) a permutation combiner providing a composition of a P permutation with a final permutation, and providing a composition of an inverse P permutation with an initial permutation.

(114) A machine according to claim 99, comprising:

(a) a mapping combiner providing a composition of a P permutation and an E expansion.

(115) A machine according to claim 99, comprising:

(a) a s-box combiner providing a composition of at least two of the following, e-expansion, s-boxes, and p permutation.

(116) A machine according to claim 99, said primitive further comprising:

(b) a folding device operative to perform folding among a plurality of results of employed form of multiplication.

(117) A machine according to claim 99, said primative further comprising:

(b) a blending device operative to perform blending among a pair of results of employed form of multiplication.

(118) A cryptobox machine comprising:

(a) a plain text provider providing a plain text.

(b) a subkey provider providing a plurality of subkeys.

(c) an inner cryptobox employing said plain text and said subkeys to generate new subkeys, and

(d) a connector providing said new subkeys to another cryptobox to process more plain texts.

(119) A machine according to claim 118 further comprising:

(e) said inner cryptobox also providing a ciphertext output, whereby encryption occurs a block at a time and information is transfered from encryption of one block to another via the new subkeys.

(120) A machine according to claim 118 further comprising:

(e) a Feistel structure with a place for a round function box.

(f) a round function box providing round-output of said new subkeys,

whose input as plaintext is said plain text from said plain text provider, and whose input as key is said subkey from said subkey provider.

(121) A machine according to claim 120 further comprising:

(g) a cryptobox key provider providing a cryptobox key.

(h) a cryptobox machine operative to encrypt said round function box output employing a cryptobox key, yielding revised round function box output employing a cryptobox key, yielding revised round function box output used in said Feistel structure.

(122) A machine according to claim 118 wherein said inner cryptobox comprises:

(a) a subkey provider providing a plurality of subkeys.

(b) a local cryptobox encrypting a master key according to encryption algorithm by using said subkey to produce a cipher text;

(c) a dependency ensurer cryptobox which ensures dependence of every bit of said cipher text on each bit of said master key by employing a plurality of said local cryptoboxes in sequence using previous cipher text as plain text:

(d) a new subkey extractor providing a single new subkey employing said local cryptobox to provide said new subkey and a new cipher text:

(c) a masterkey mixer employing a series of local cryptoboxes, at least zero times in series using previous cipher text as plain text.

(123) A machine according to claim 118 comprising:

(a) employing said inner cryptobox a pluraltiy of times, using new subkeys of previous as subkey of next time, using cipher text of previous as plain text of next time, yielding new subkeys of final time as output.

(124) A method for automatically protecting confidentiality of information stored on a persistent storage medium, the informatin being organized into a plurality of files, the steps of the method comprising:

(a) protecting a plurality of files on an automatic file-by-file basis such that each of said plurality of files is automatically protected individually according to the steps of:

(i) using a cryptosystem to encrypt said at least one file, thereby generating an encrypted file: and

(ii) storing said encrypted file on the persistent storage medium.

(125) The method according to claim 124, wherein the persistent storage medium is operated by a computation device having a sleep mode, said computational device having a RAM (random access memory), such that all information in said RAM is encrypted and written to the persistent storage medium as a unit.

(126) The method according to claim 124, wherein the persistent storage medium is operated by a computational device having a sleep mode, said computational device having a RAM (random access memory), such that at least a portion of information in said RAM is encrypted.

(127) The method according to claim 124, wherein a ramdisk is used to store key material.

(128) A method according to claim 124, comprising the steps of:

(129) A method according to claim 124, comprising the steps of:

(b) performing a portion of said cryptosystem using a logic-gate implementation of a bit-slice representation of at least one component of said cryptosystem.

(130) A method according to claim 124, said cryptosystem comprising the steps of:

(a) providing plain text

(b) providing a plurality of subkeys,

(c) employing said plain text and said subkeys to generate new subkeys, and

(d) providing said new subkeys to another cryptosystem to process more plain texts.

Figure 1

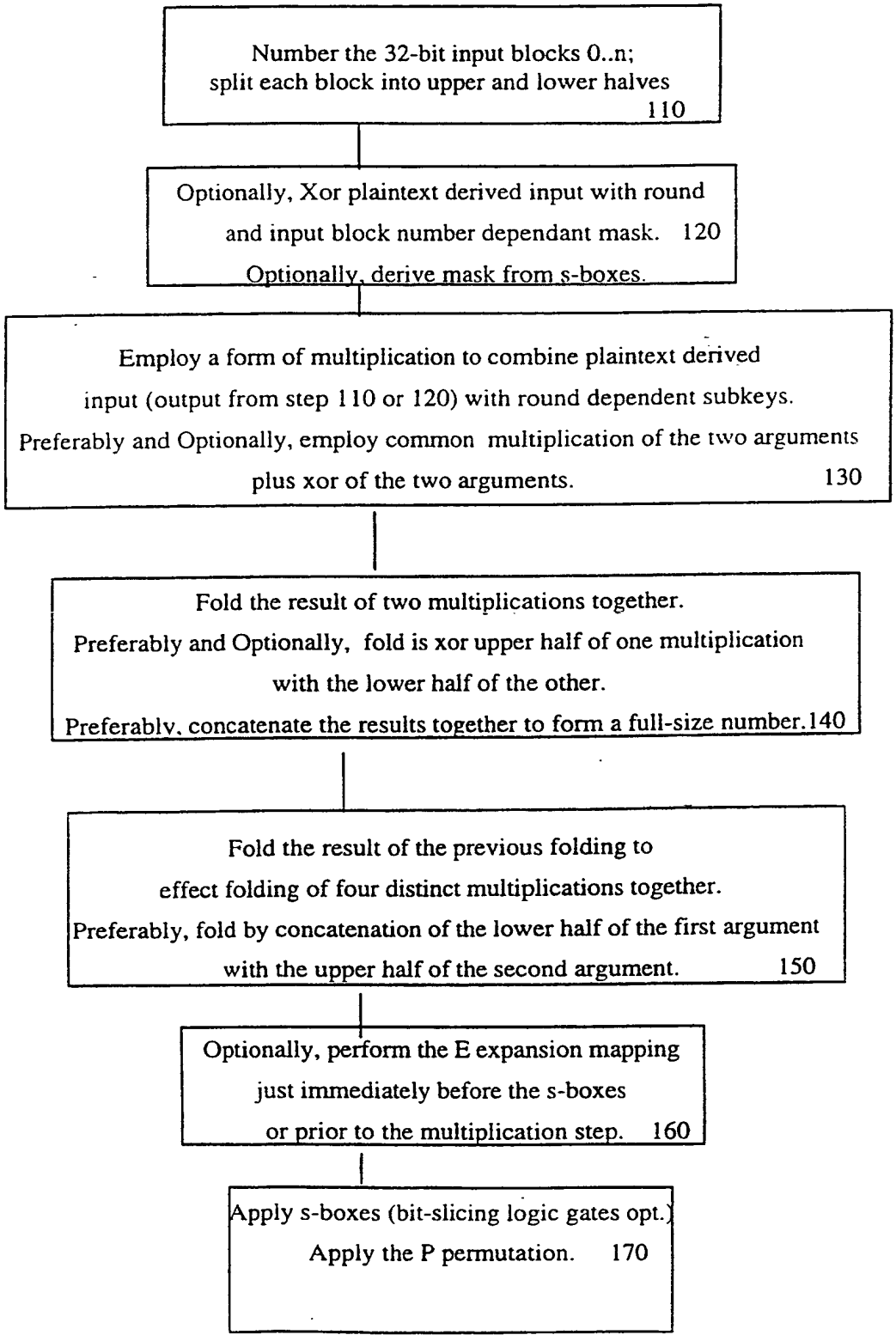


Figure 2

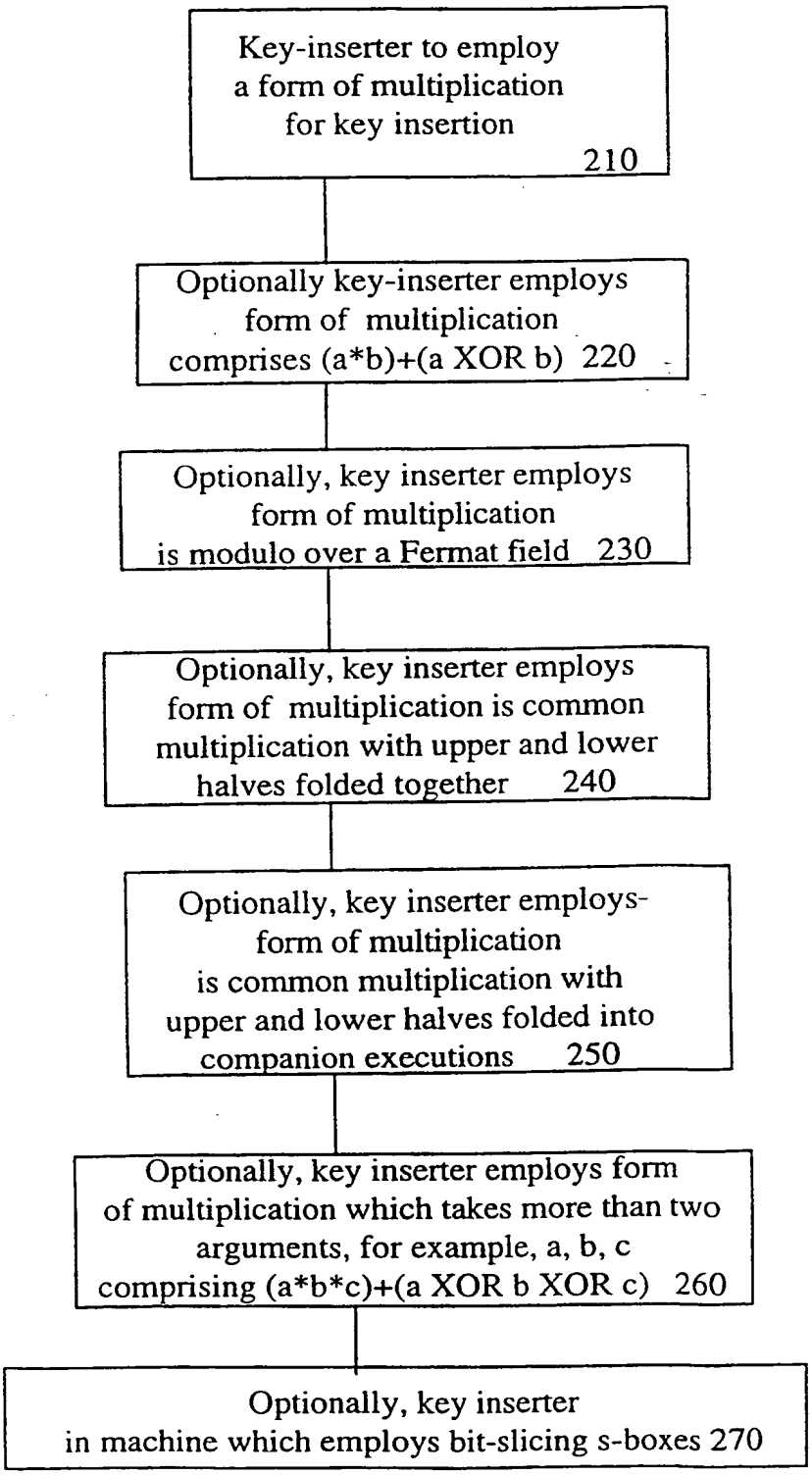


Figure 3

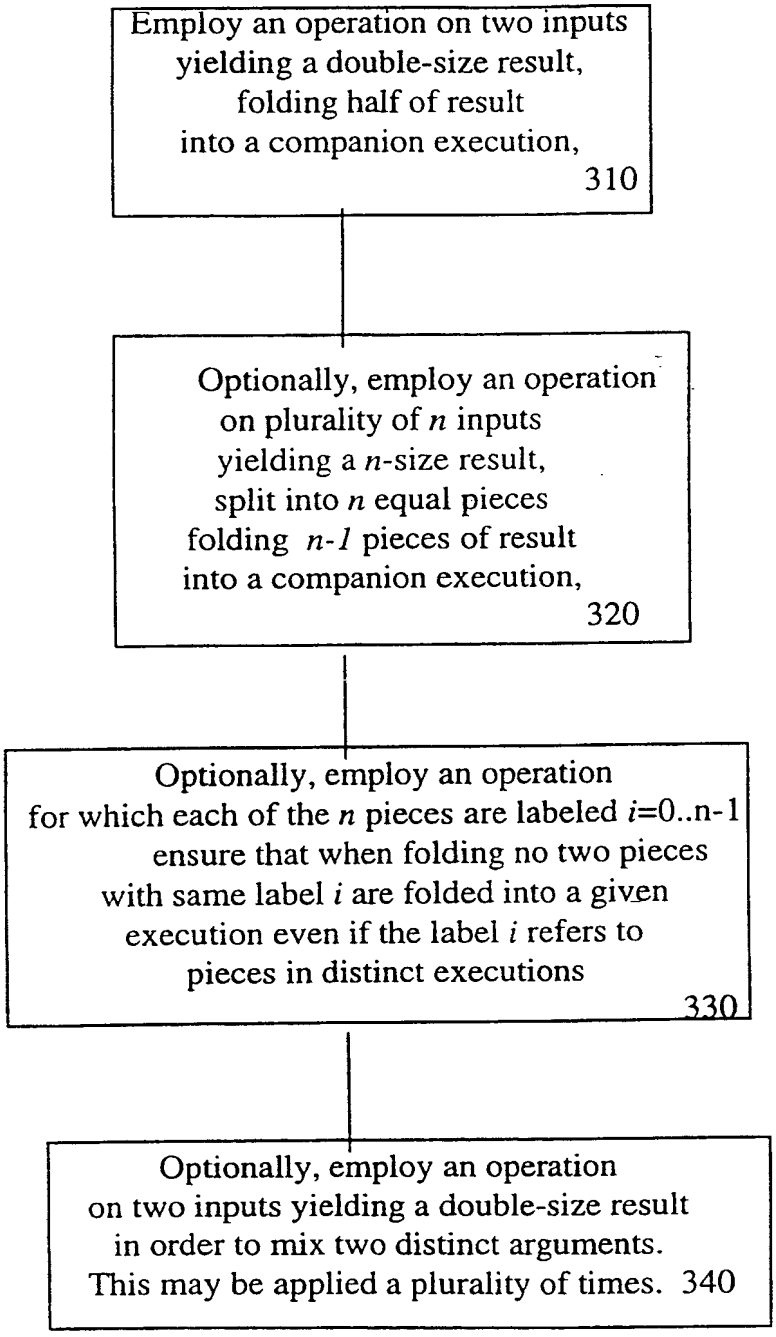


Figure 4

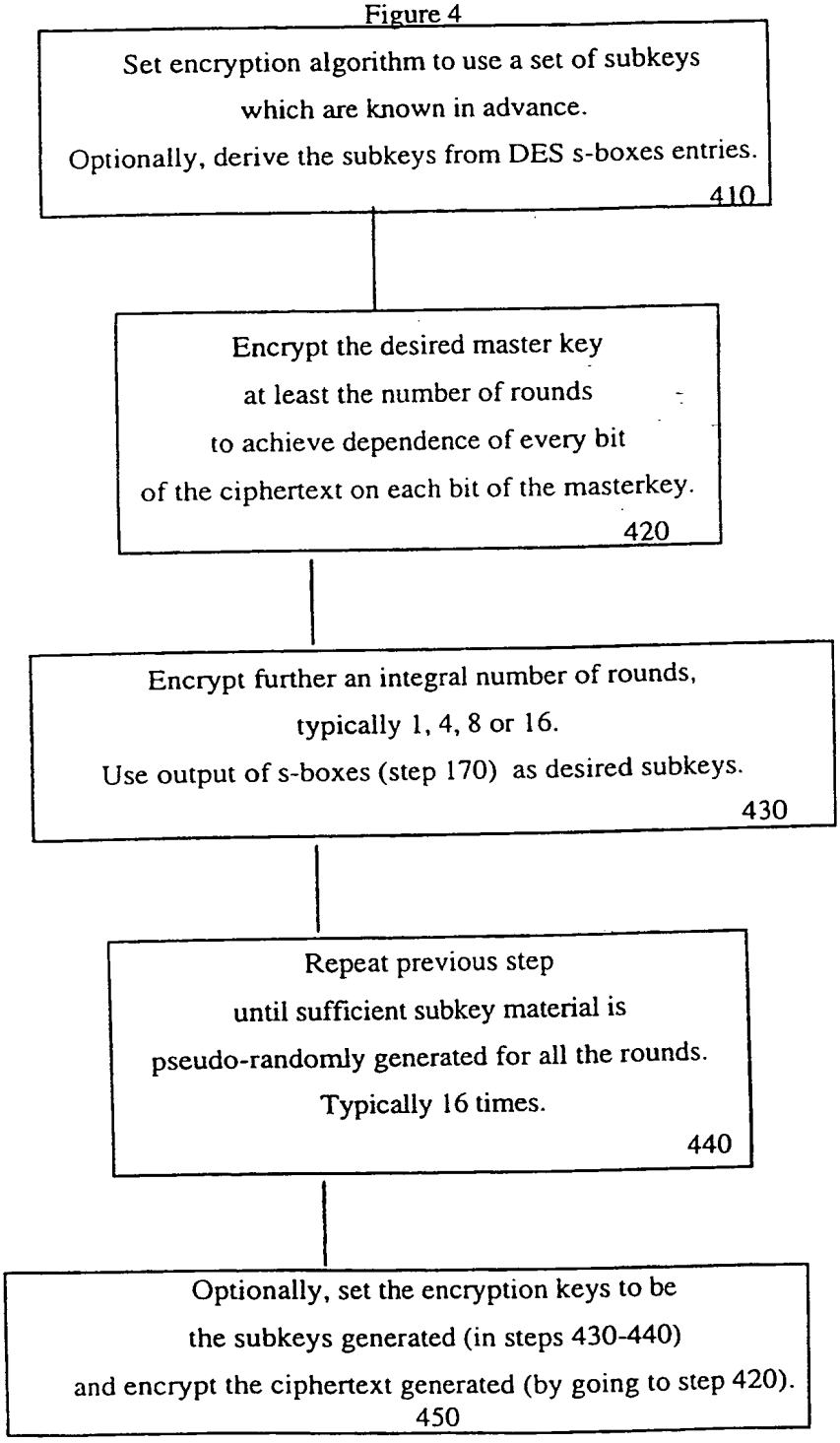


Figure 5

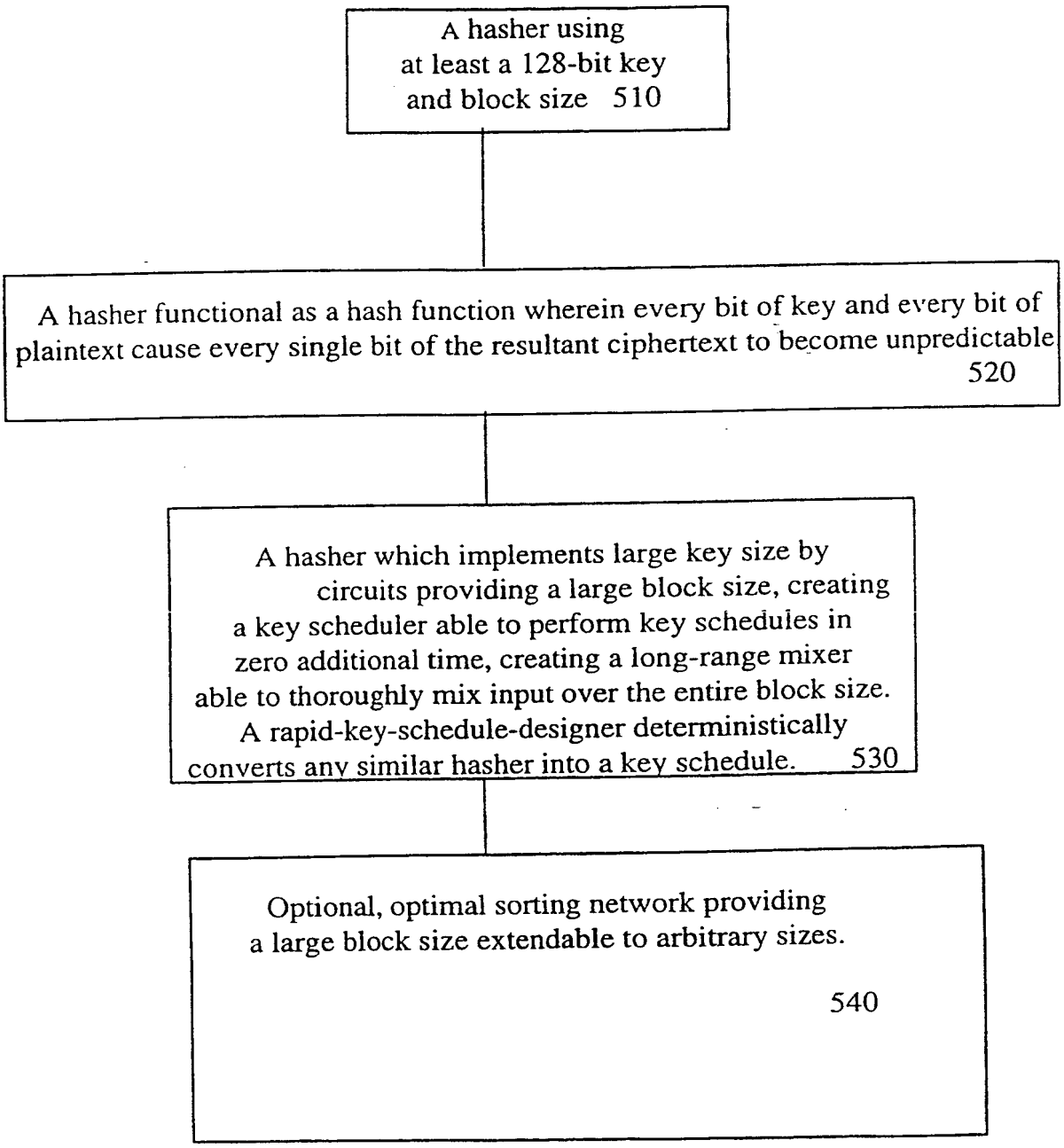


Figure 6

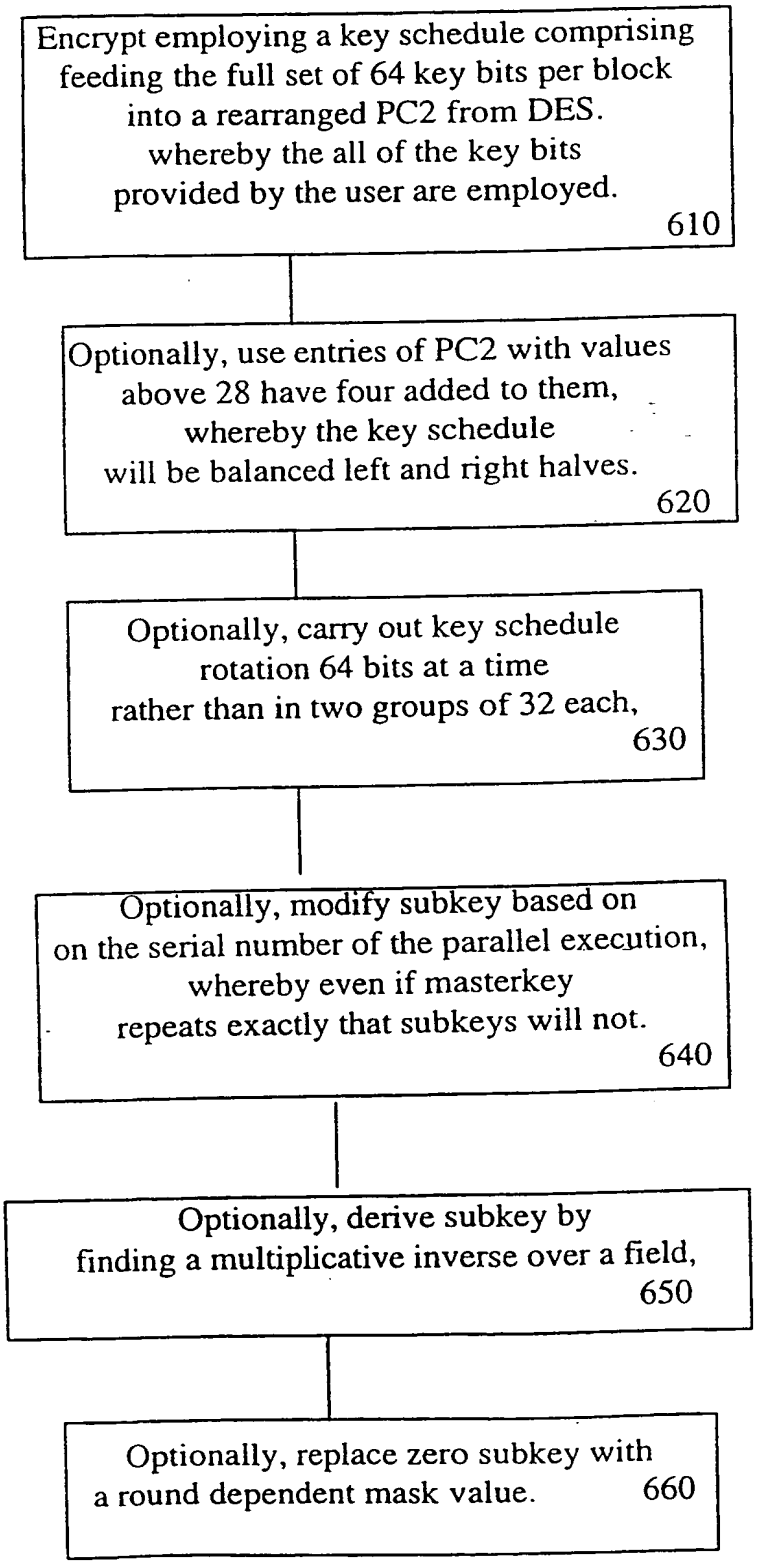


Figure 7

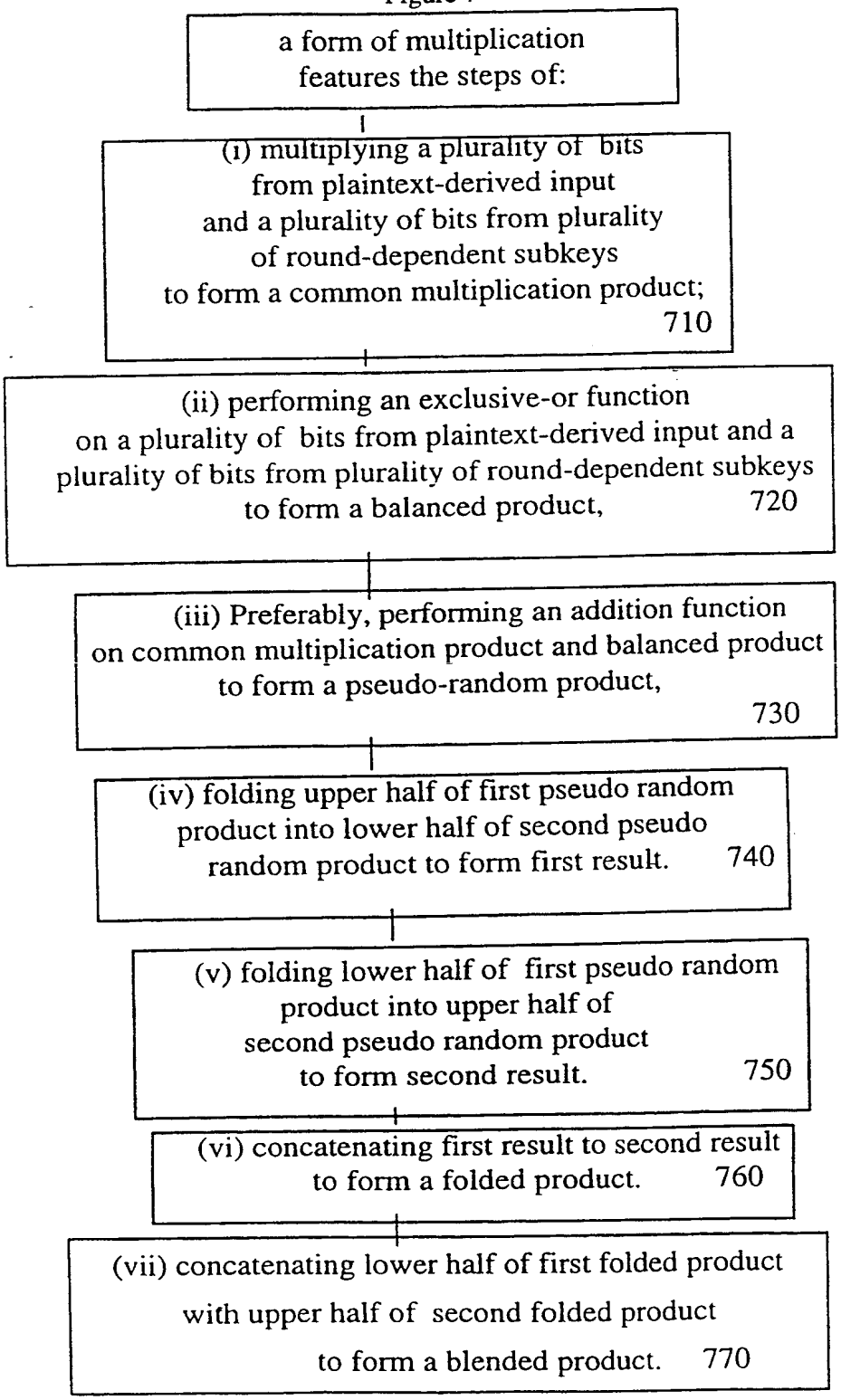


Figure 8

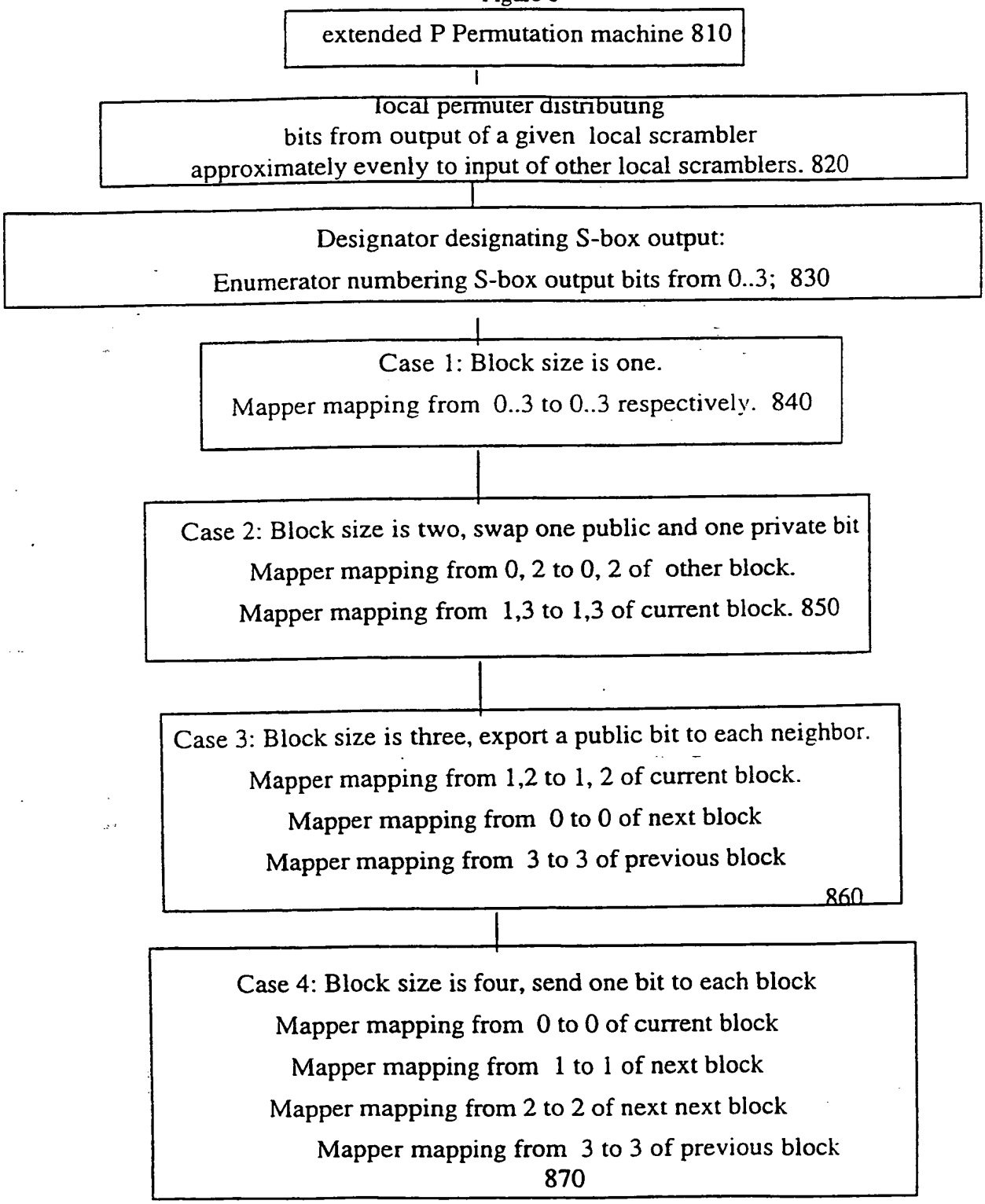


Figure 9

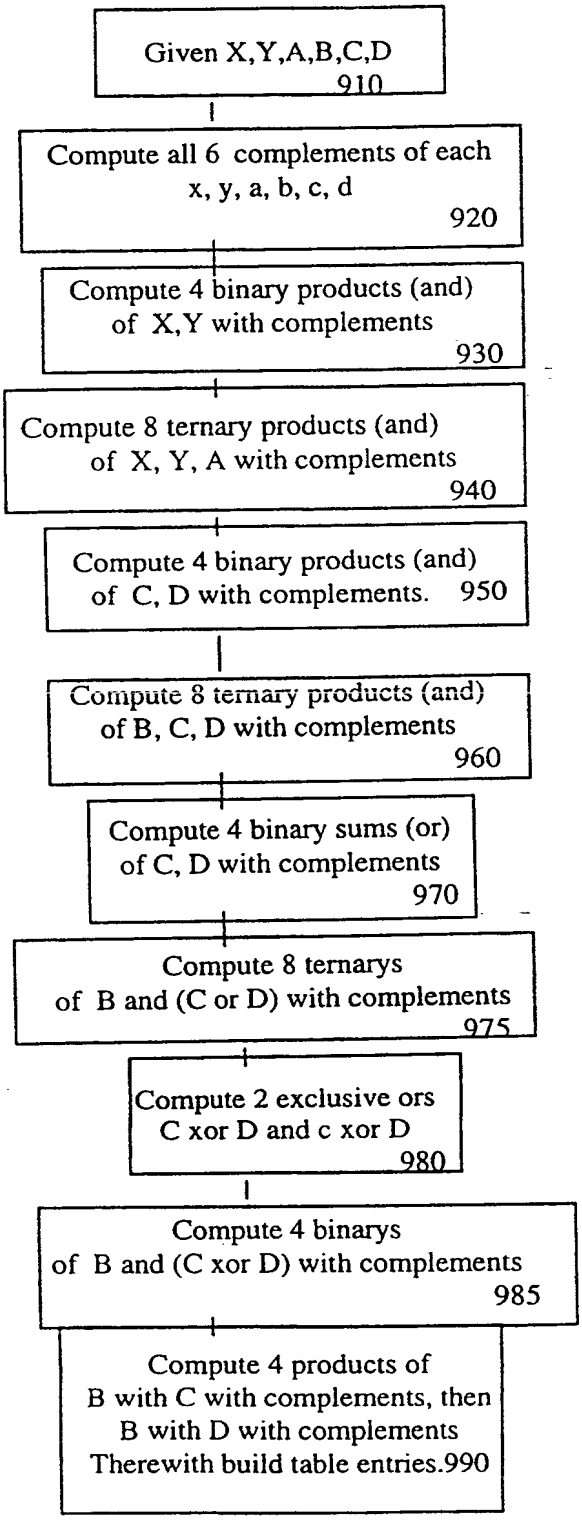


Figure 10

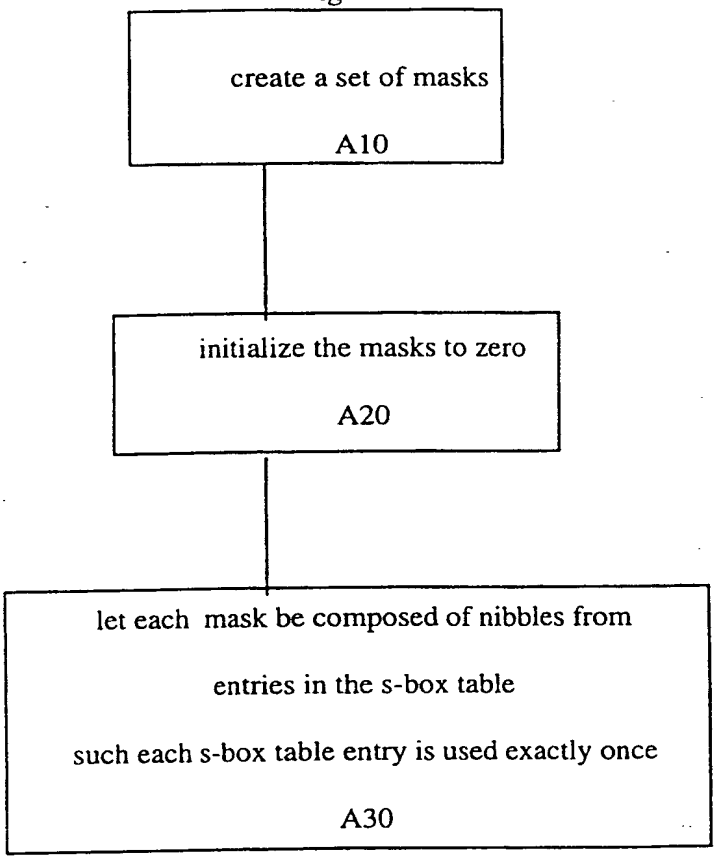



**TABLE I**
**Key Selection Permutation Table**

| | | | | | |
|---|---|---|---|---|---|
| 14 | 17 | 11 | 24 | 1 | 5 |
| 45 | 56 | 35 | 41 | 51 | 59 |
| 3 | 28 | 15 | 6 | 21 | 10 |
| 48 | 53 | 43 | 60 | 38 | 57 |
| 23 | 19 | 12 | 4 | 26 | 8 |
| 34 | 44 | 55 | 49 | 37 | 52 |
| 16 | 7 | 27 | 20 | 13 | 2 |
| 50 | 46 | 54 | 40 | 33 | 36 |

Figure 11

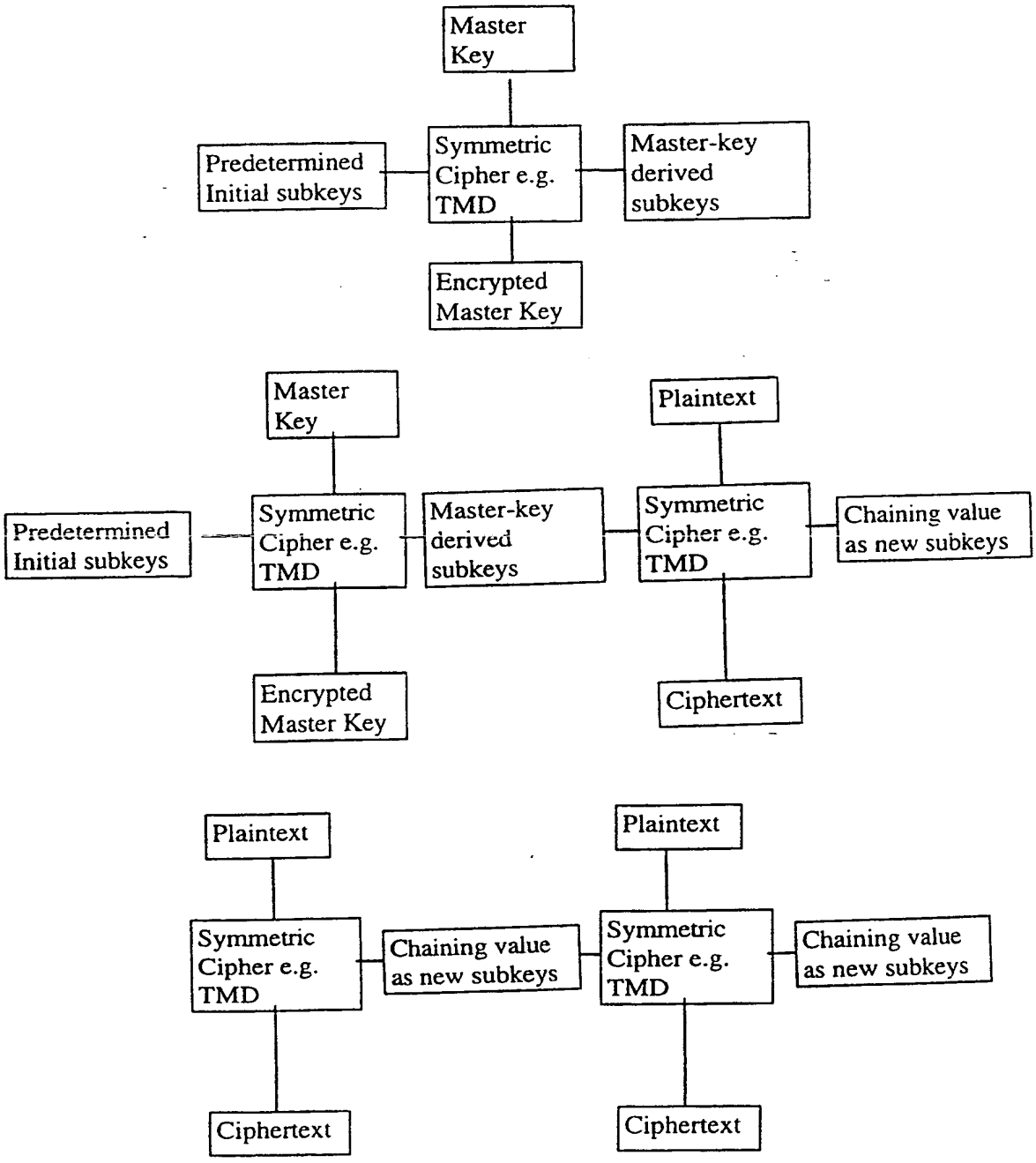


Figure 12

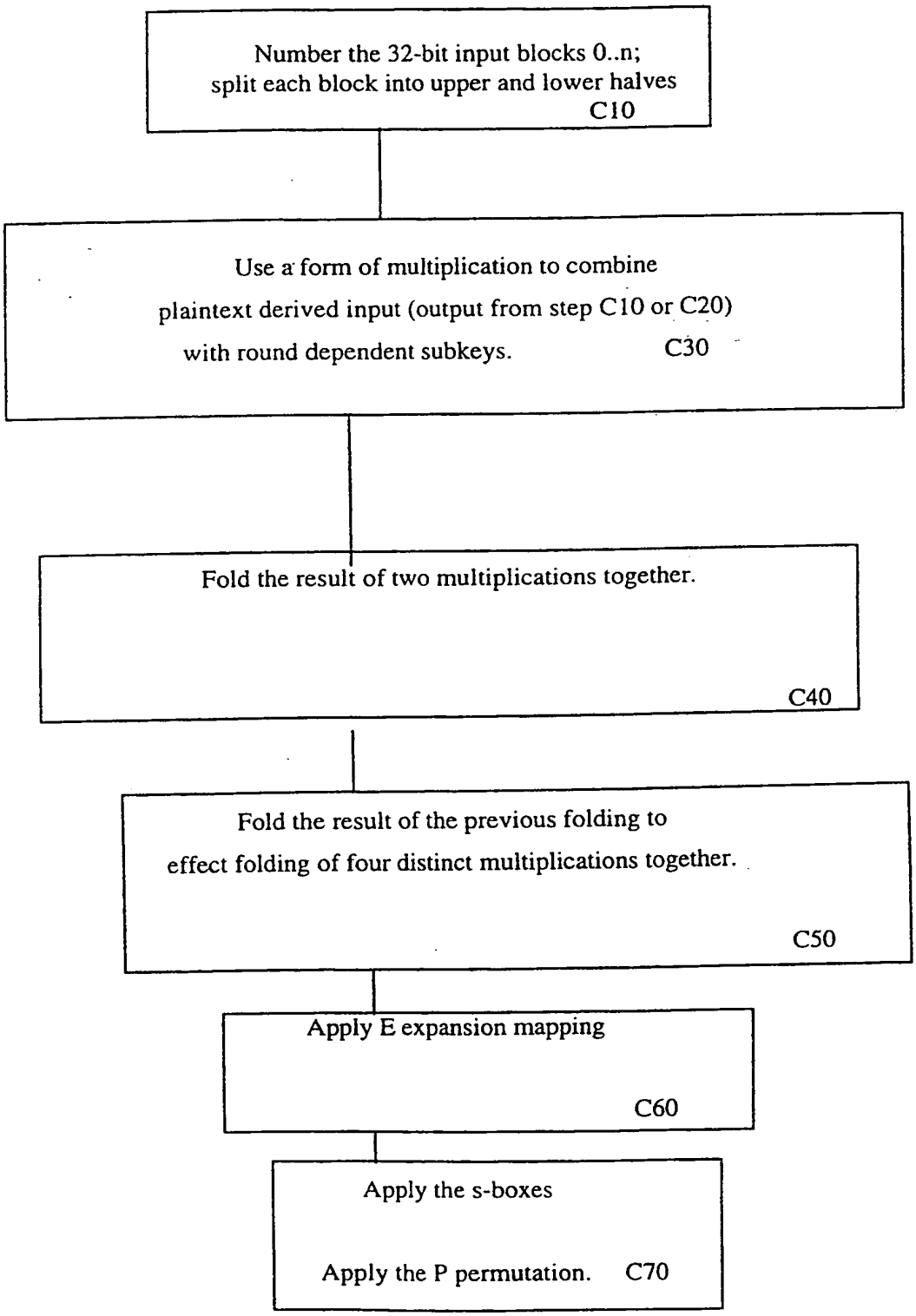


Figure 13

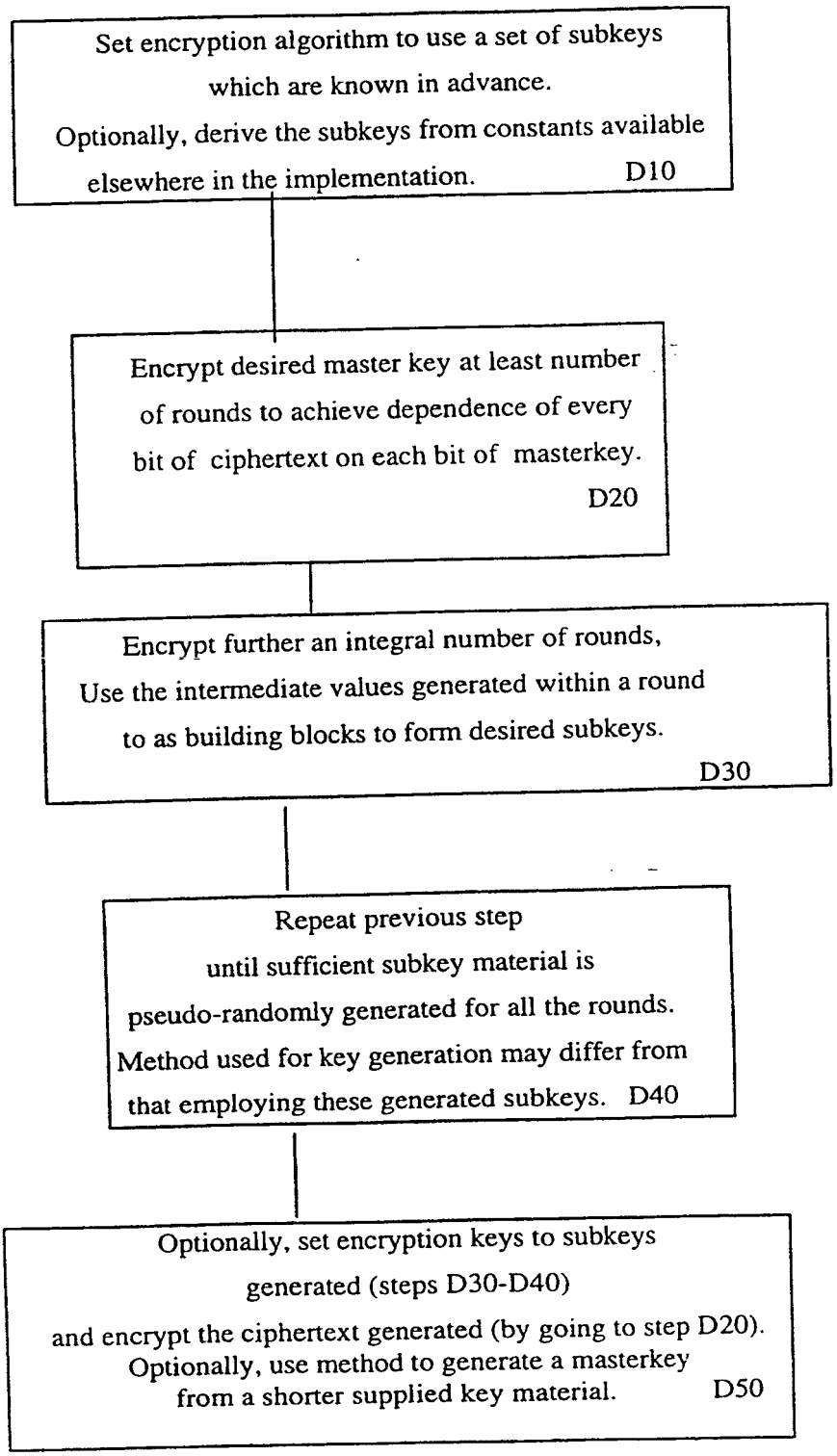
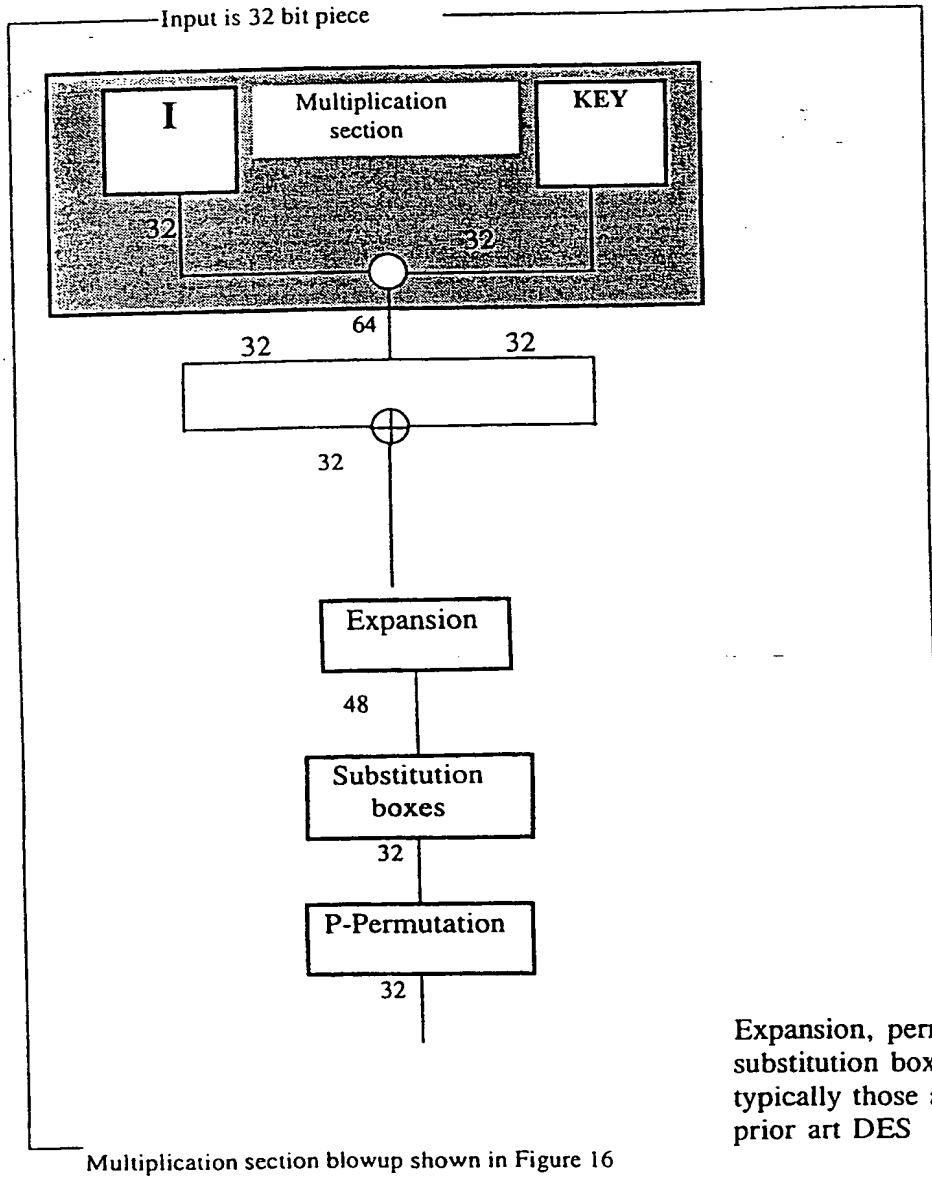


Figure 14

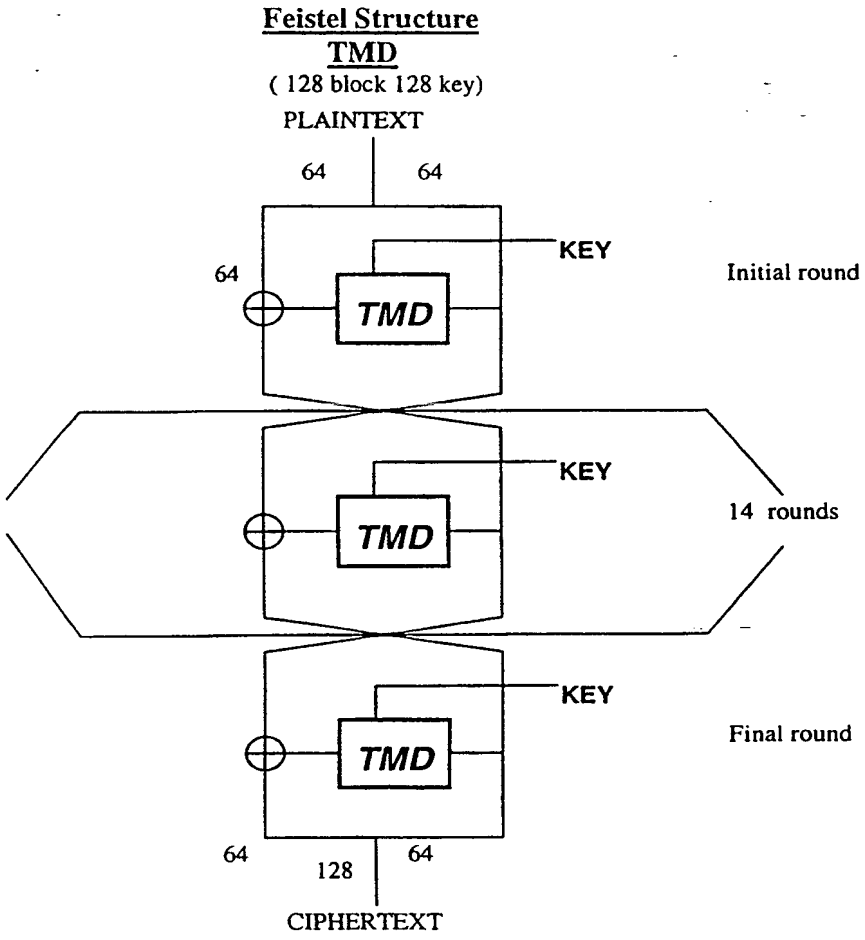
# MultiDES
## INTERNAL ROUND

(64 block size, 64 key size)




Expansion, permutation and substitution boxes shown are typically those appearing in prior art DES

Figure 15




Legend for figures 14-16:
(+) indicates exclusive-or
( ) indicates a form of multiplication
□ with a "+" inside is standard addition

Figure 16

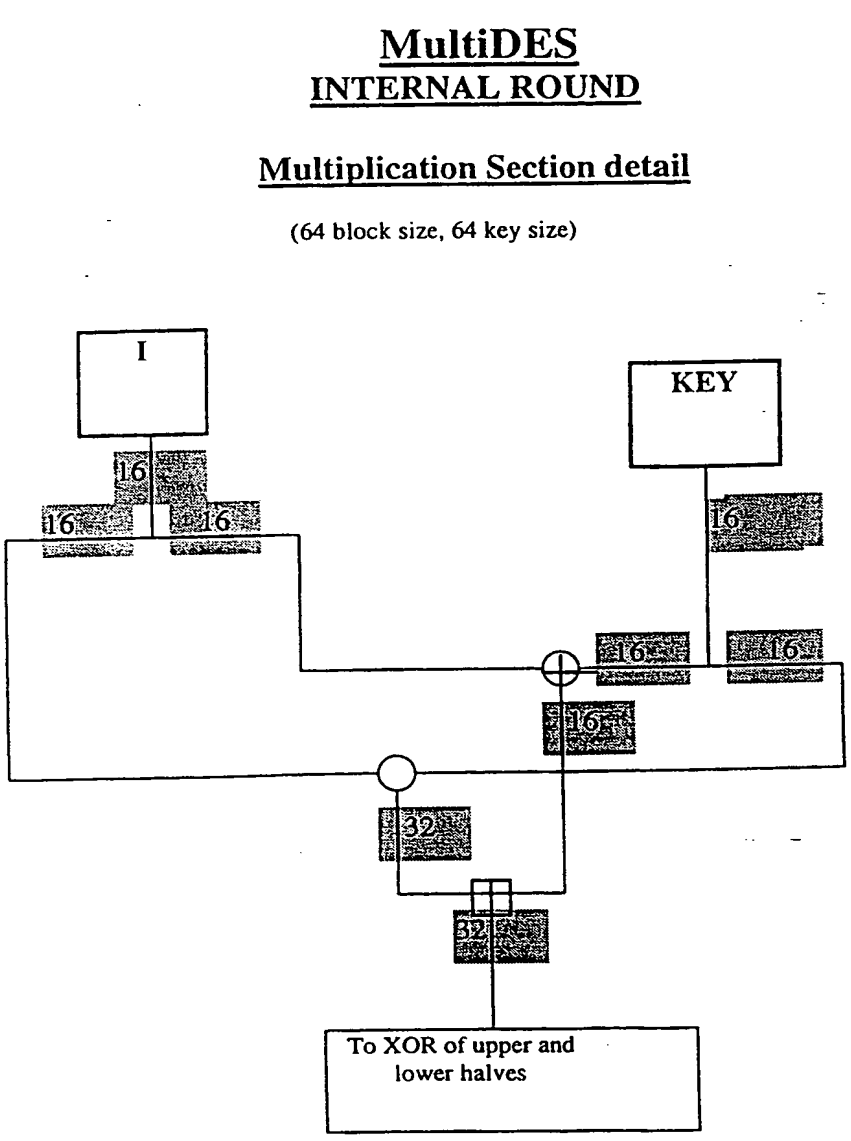


Figure 17

Legend figures 17-19:
( ) receives 32-bit inputs, yields 64-bit output
(+) receives 32-bits inputs, yields 32-bit output

**TMD: INTERNAL ROUND**
**Two MultiDES Rounds**
(128 block size, 128 key size)

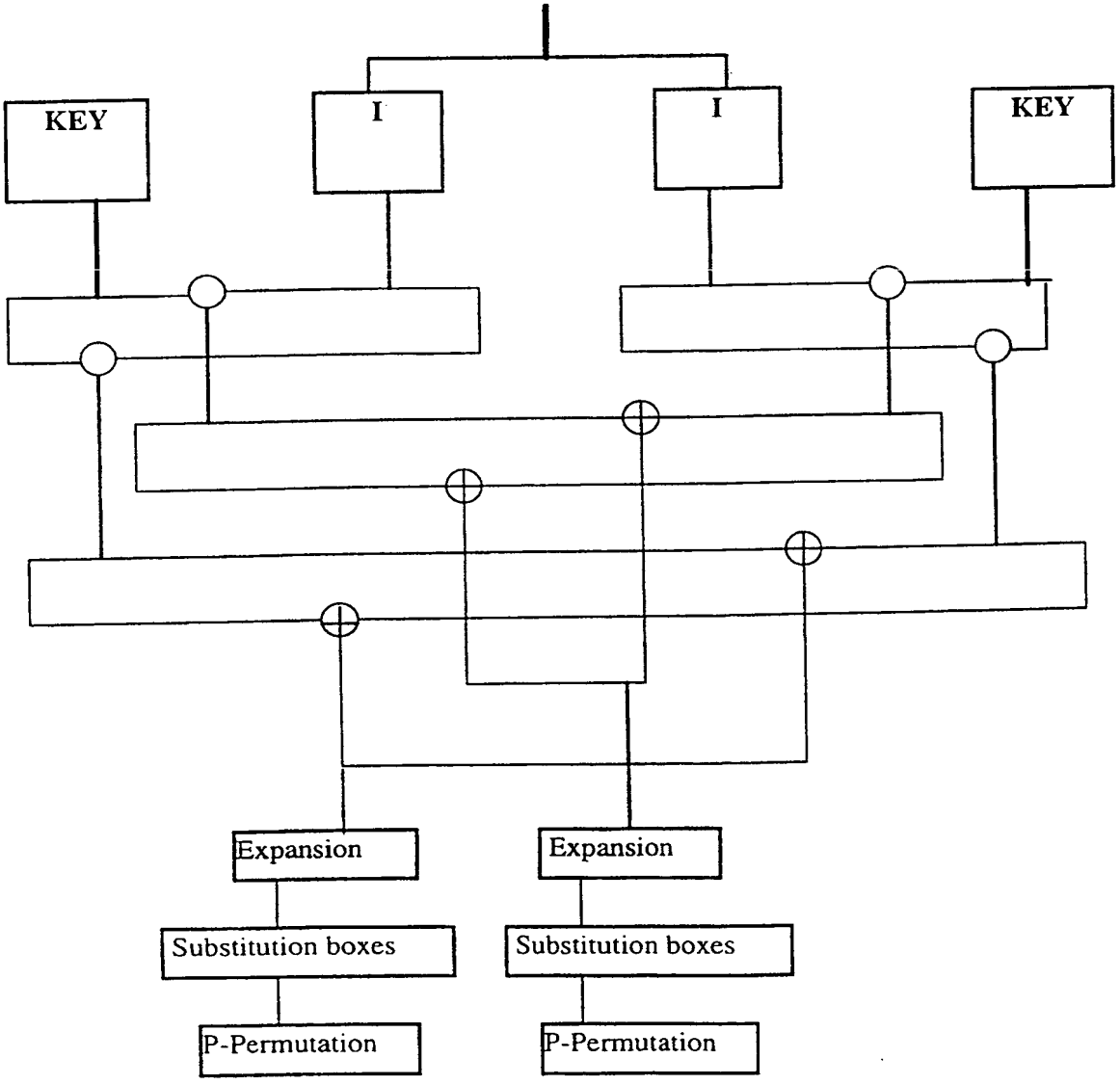


Figure 18

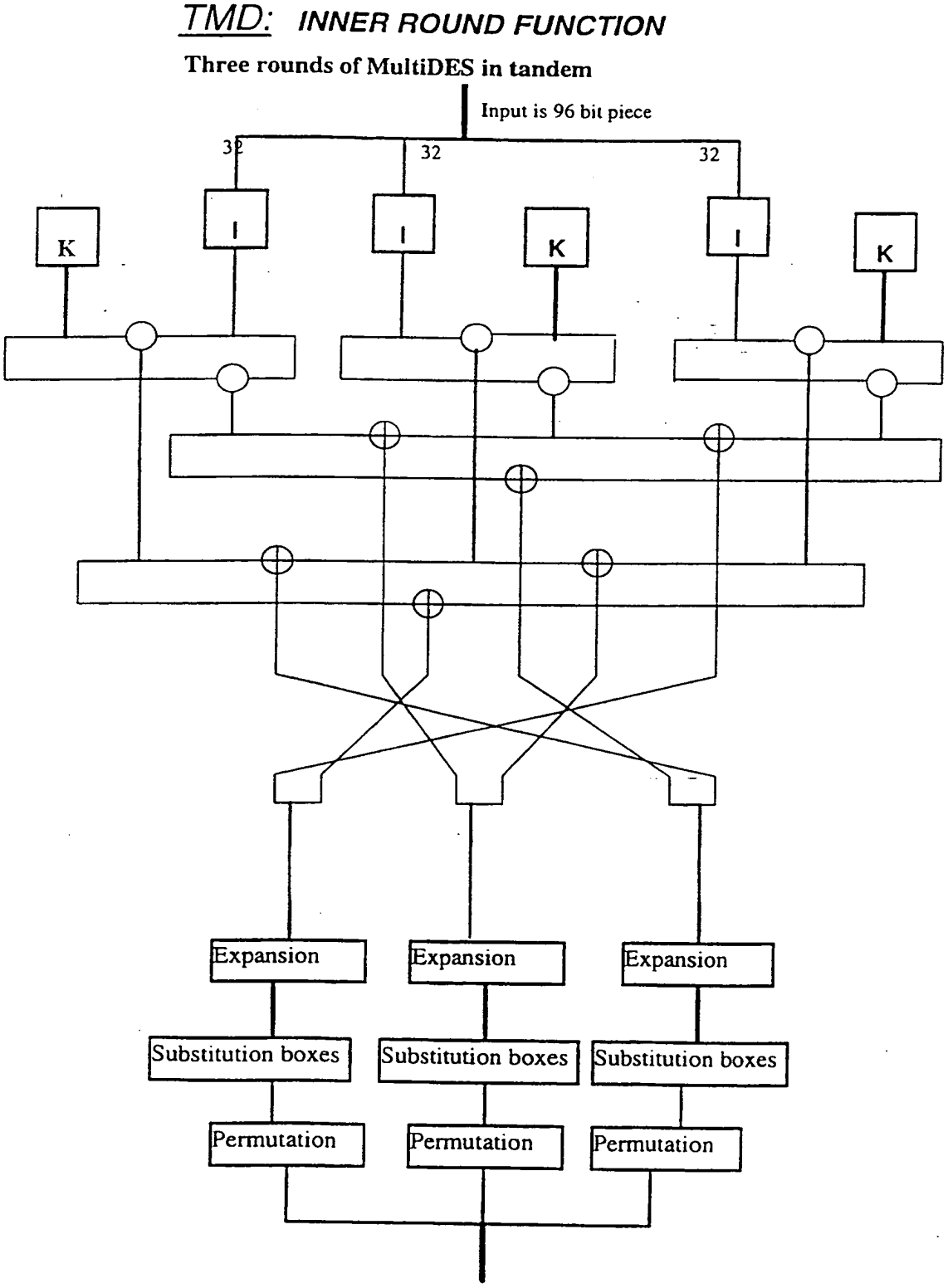


Figure 19

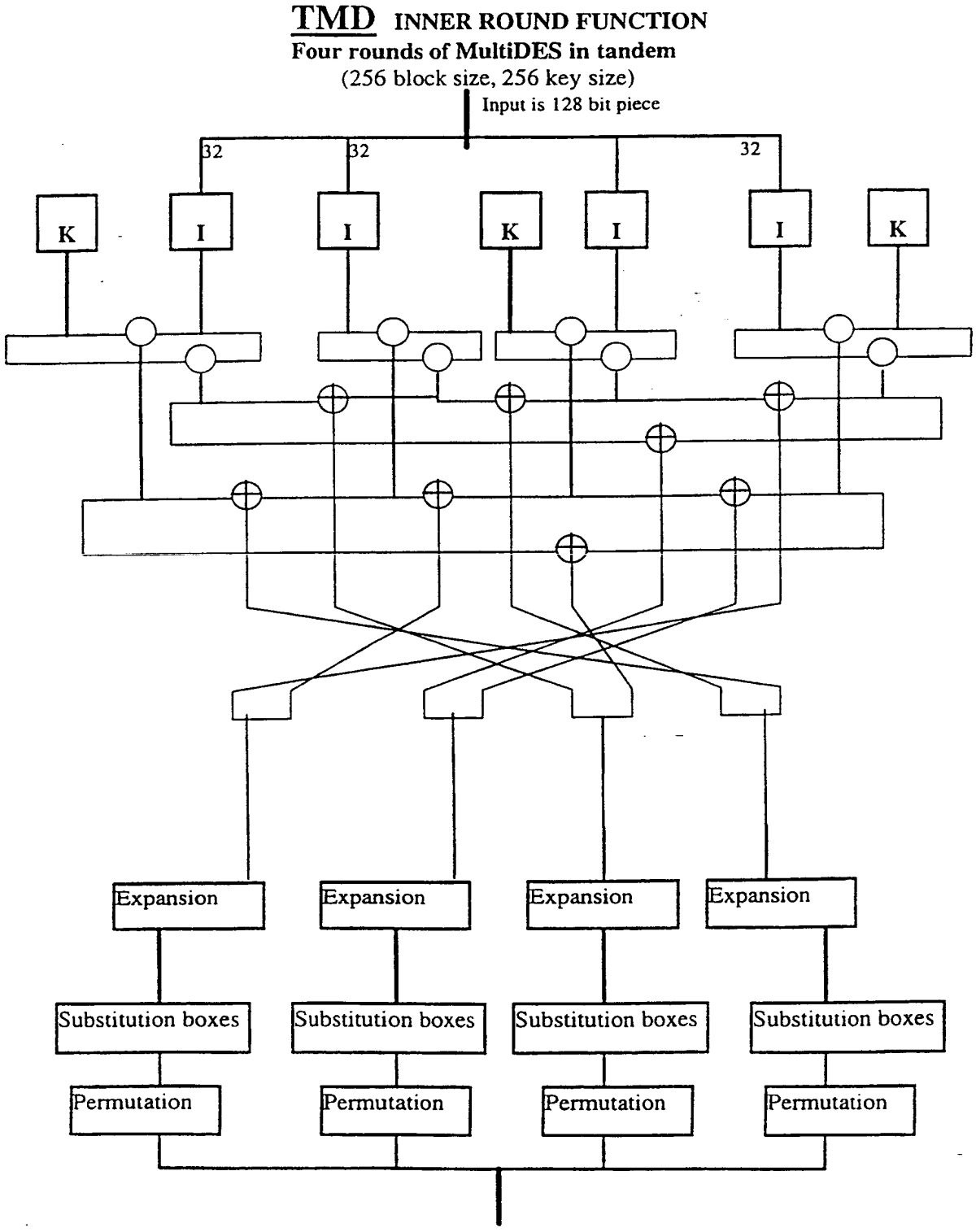


Figure 20

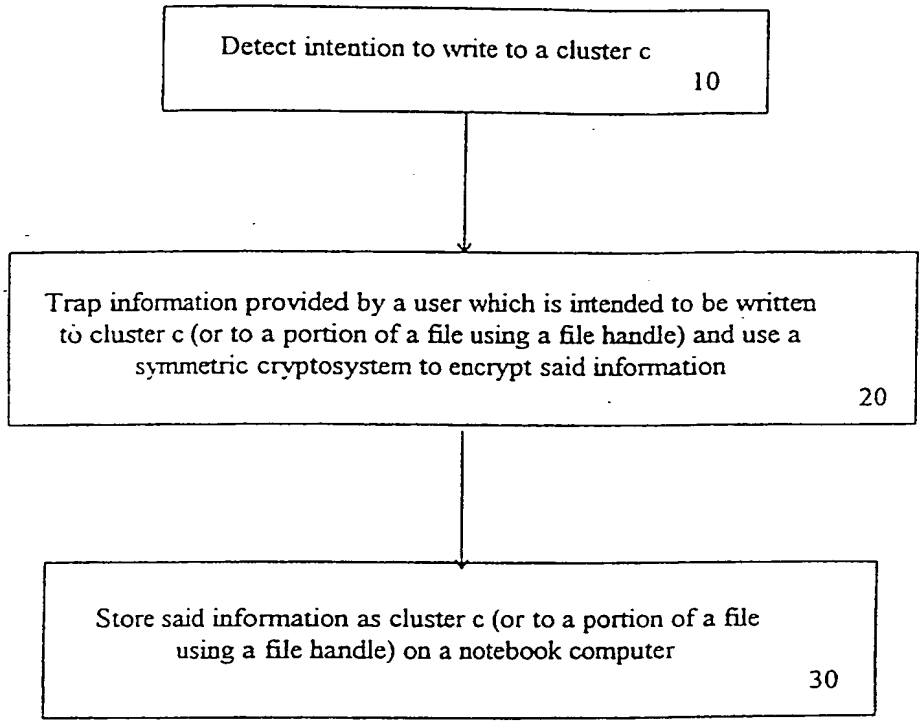


Figure 21

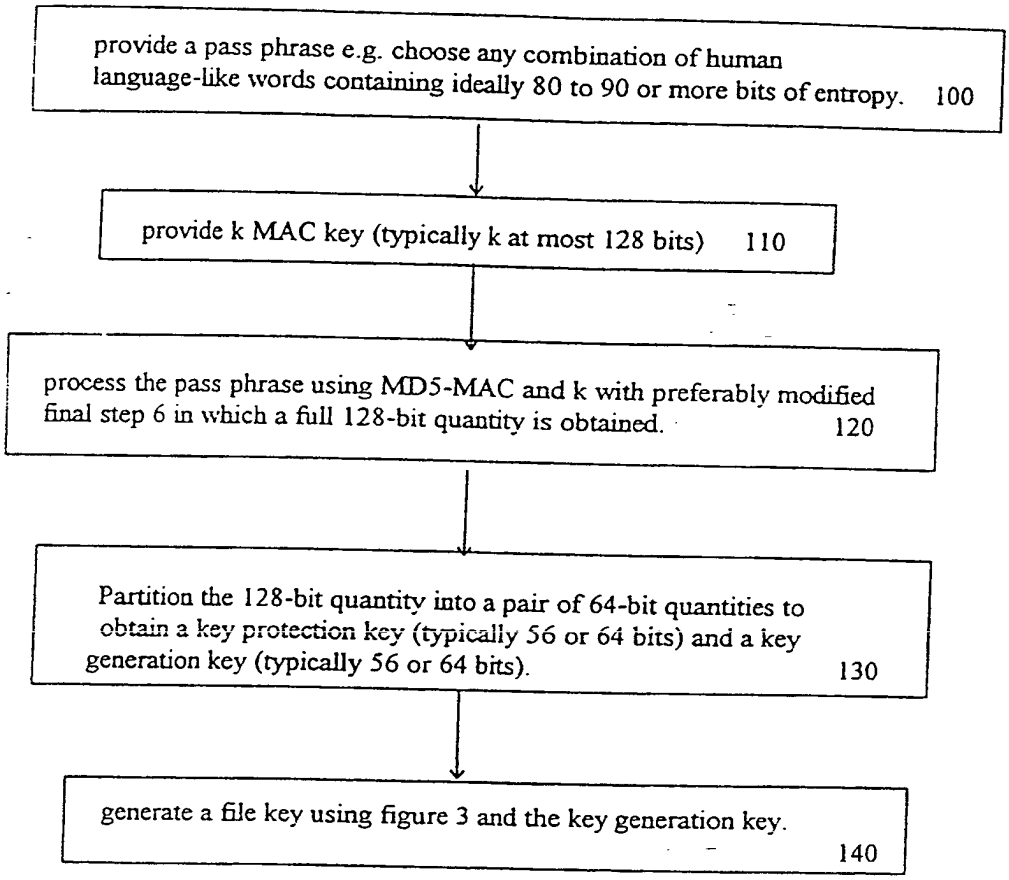


Figure 22

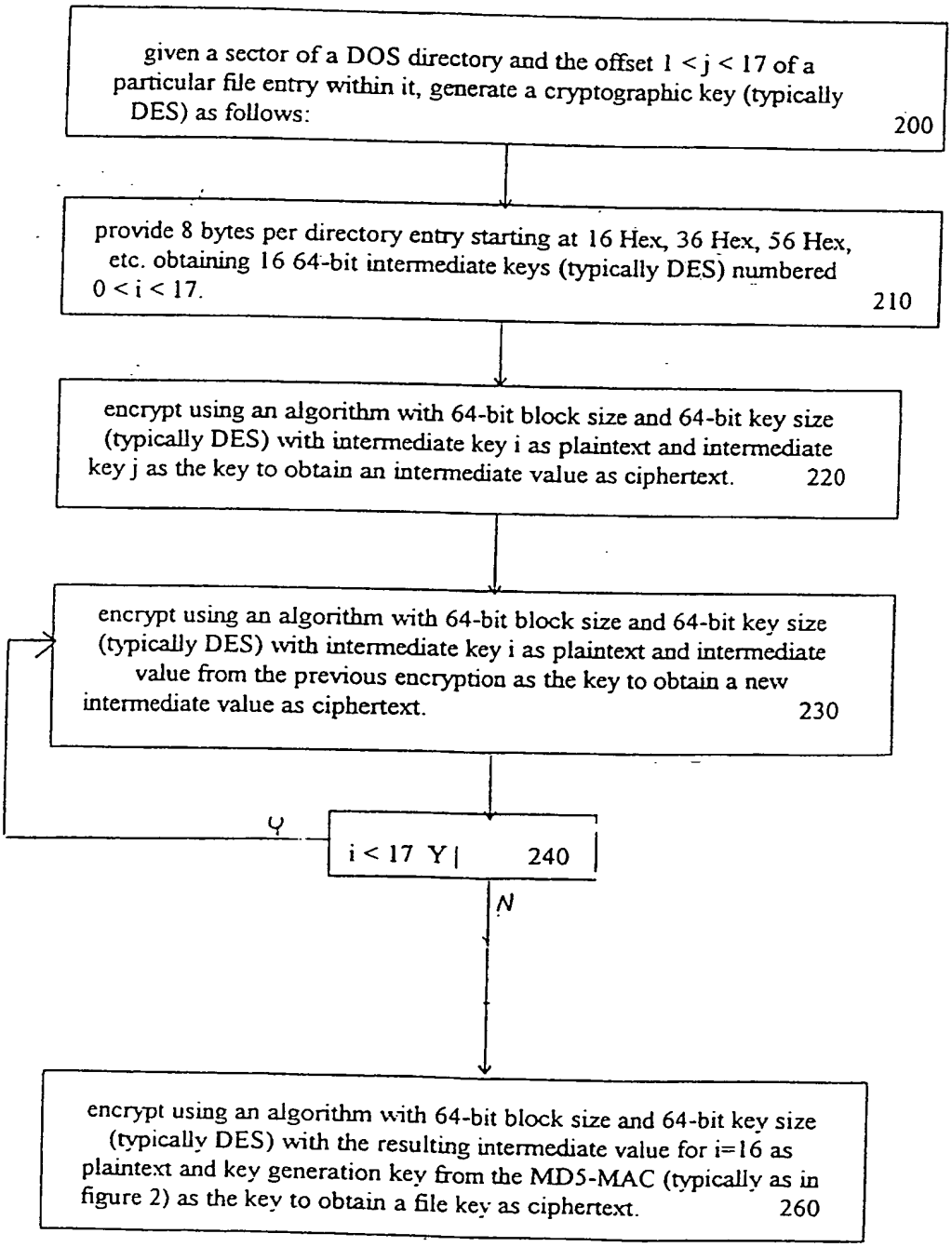


Figure 23

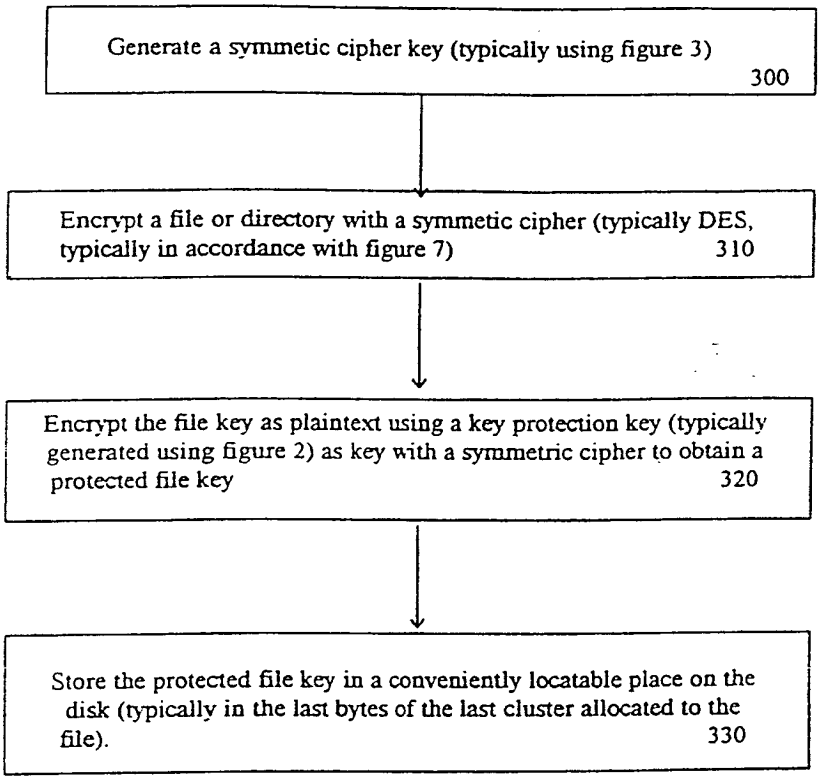


Figure 24

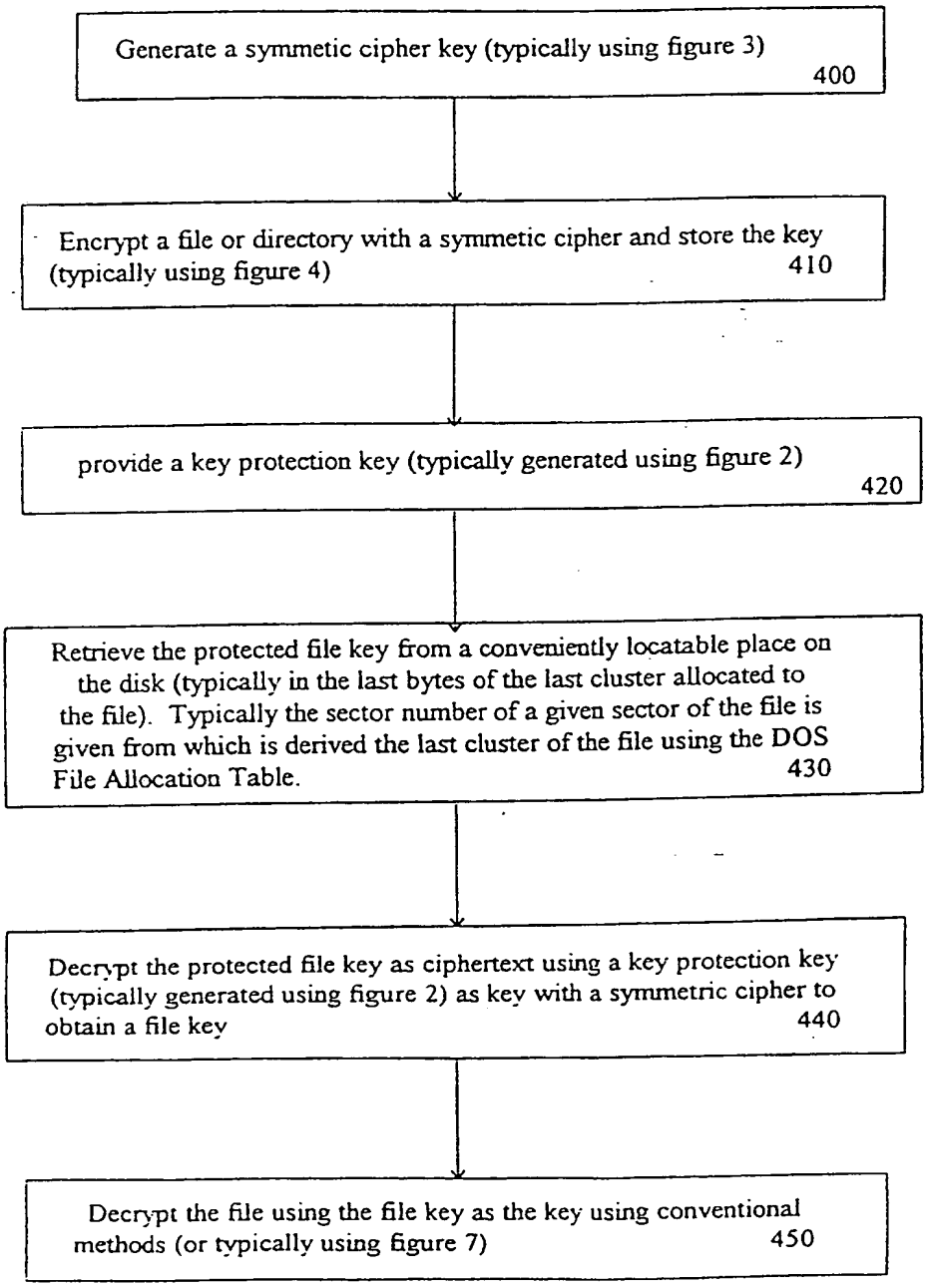


Figure 25

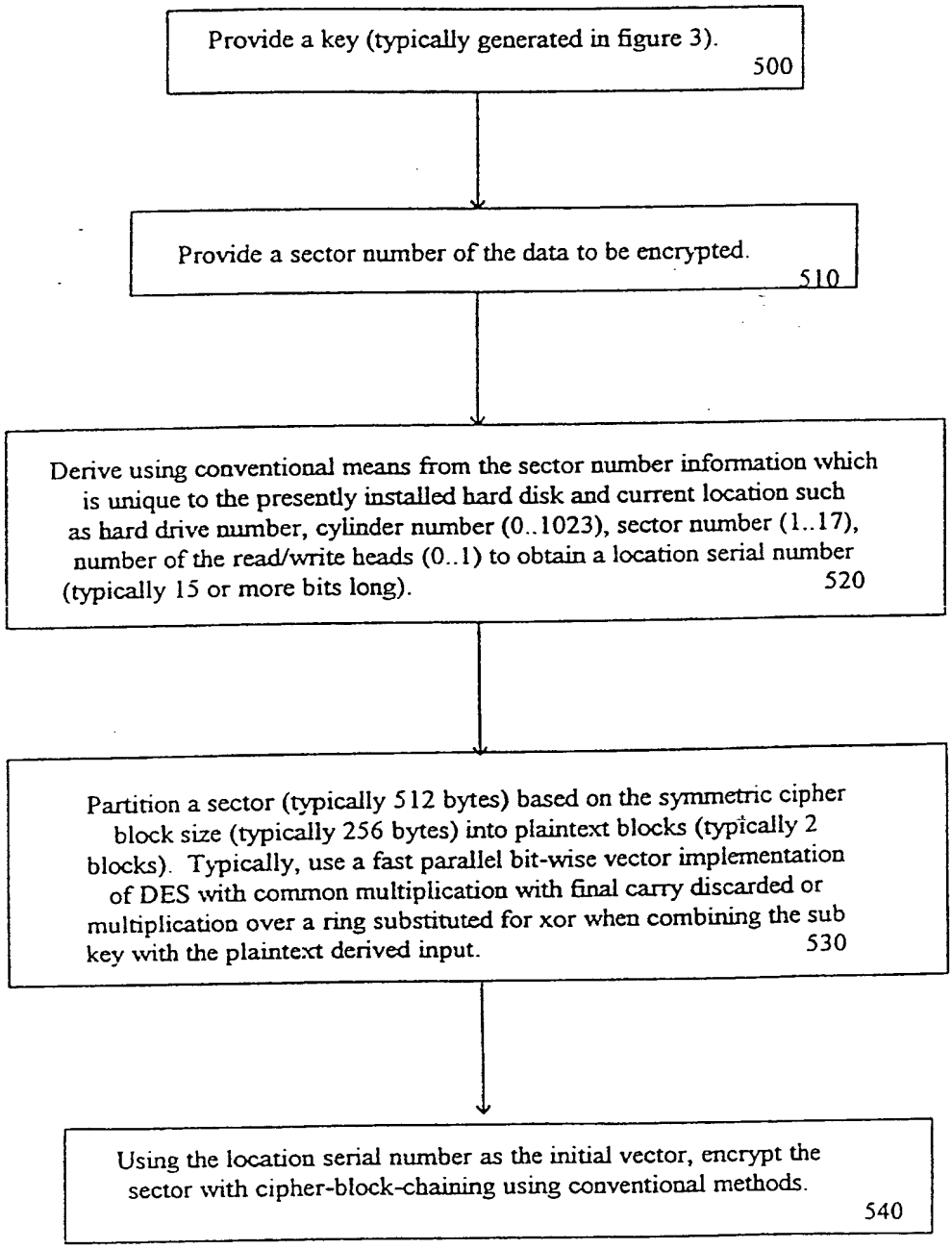


Figure 26

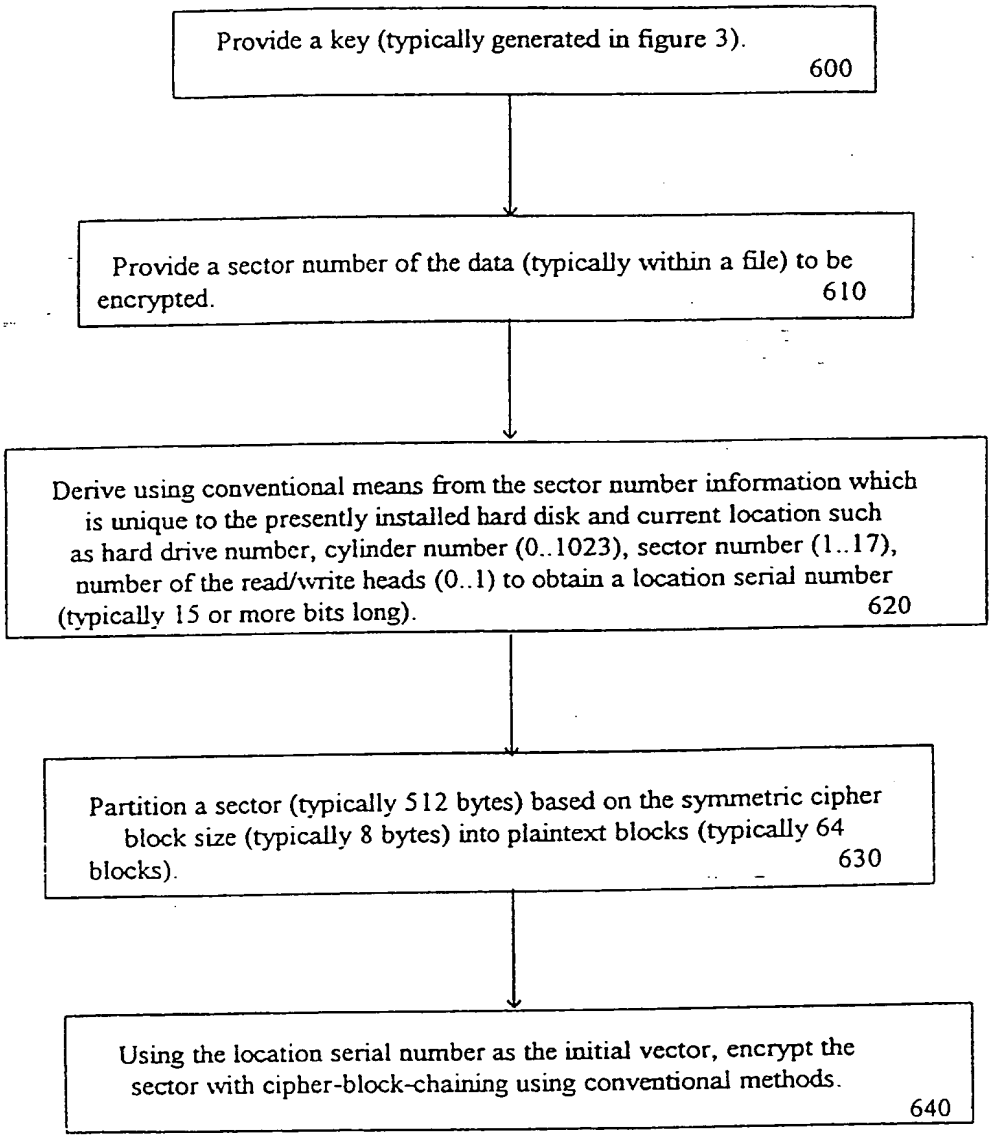


Figure 27

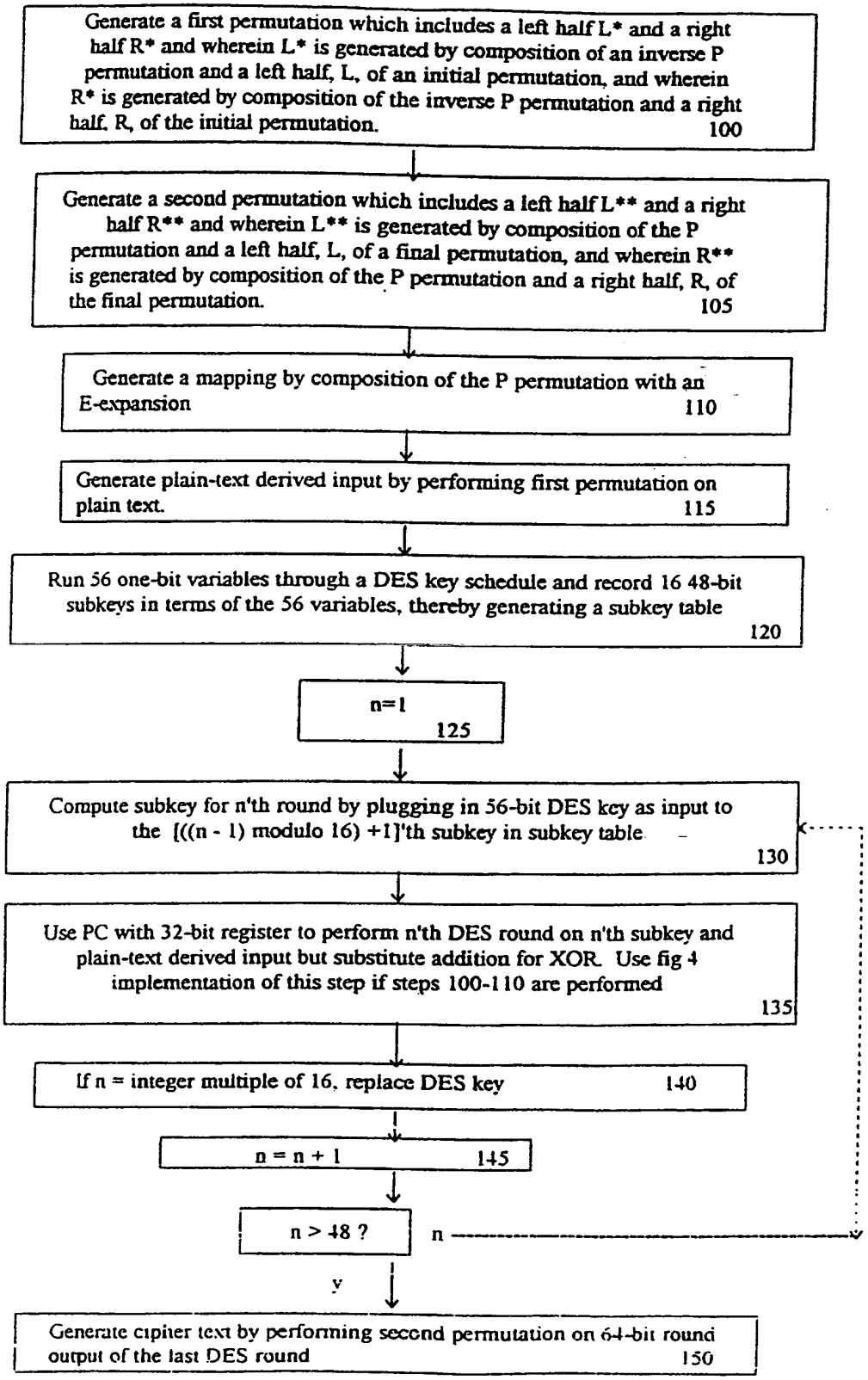


Figure 28

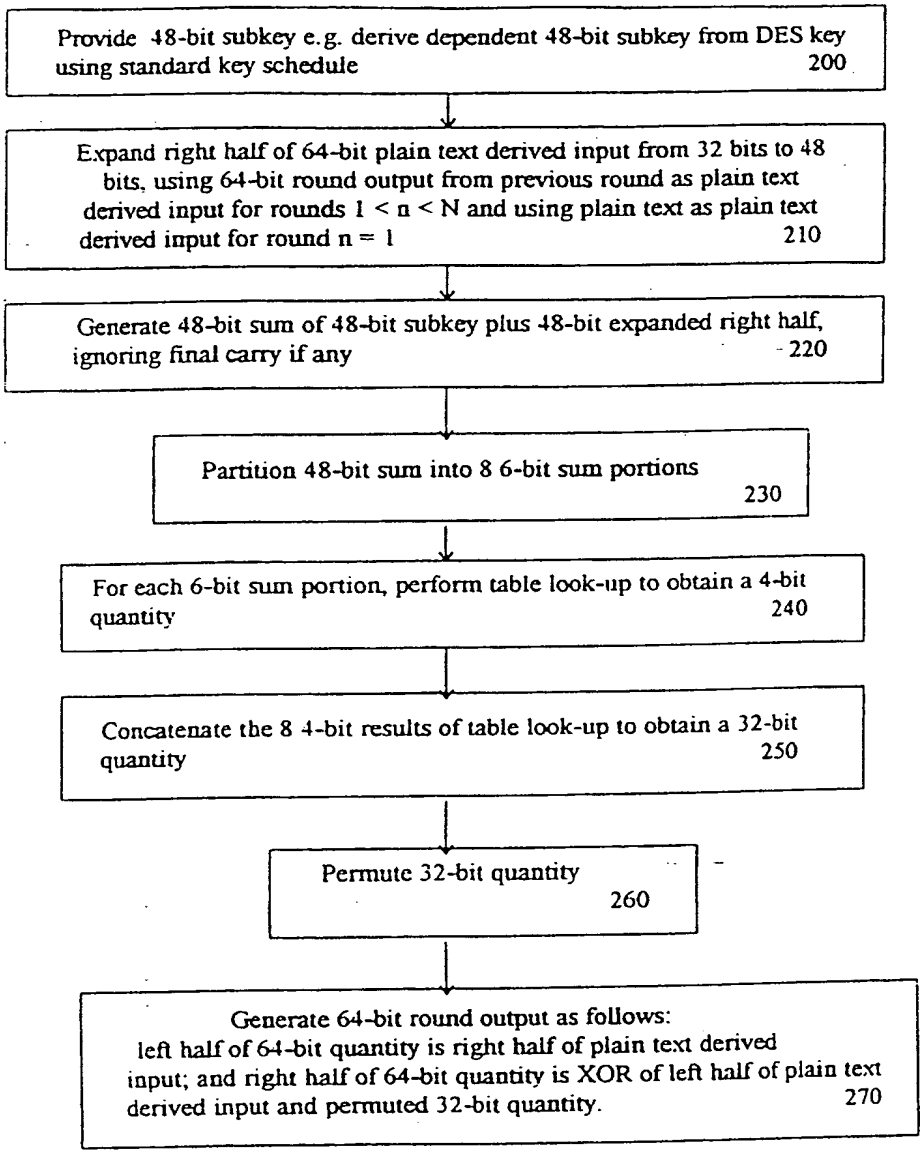


Figure 29

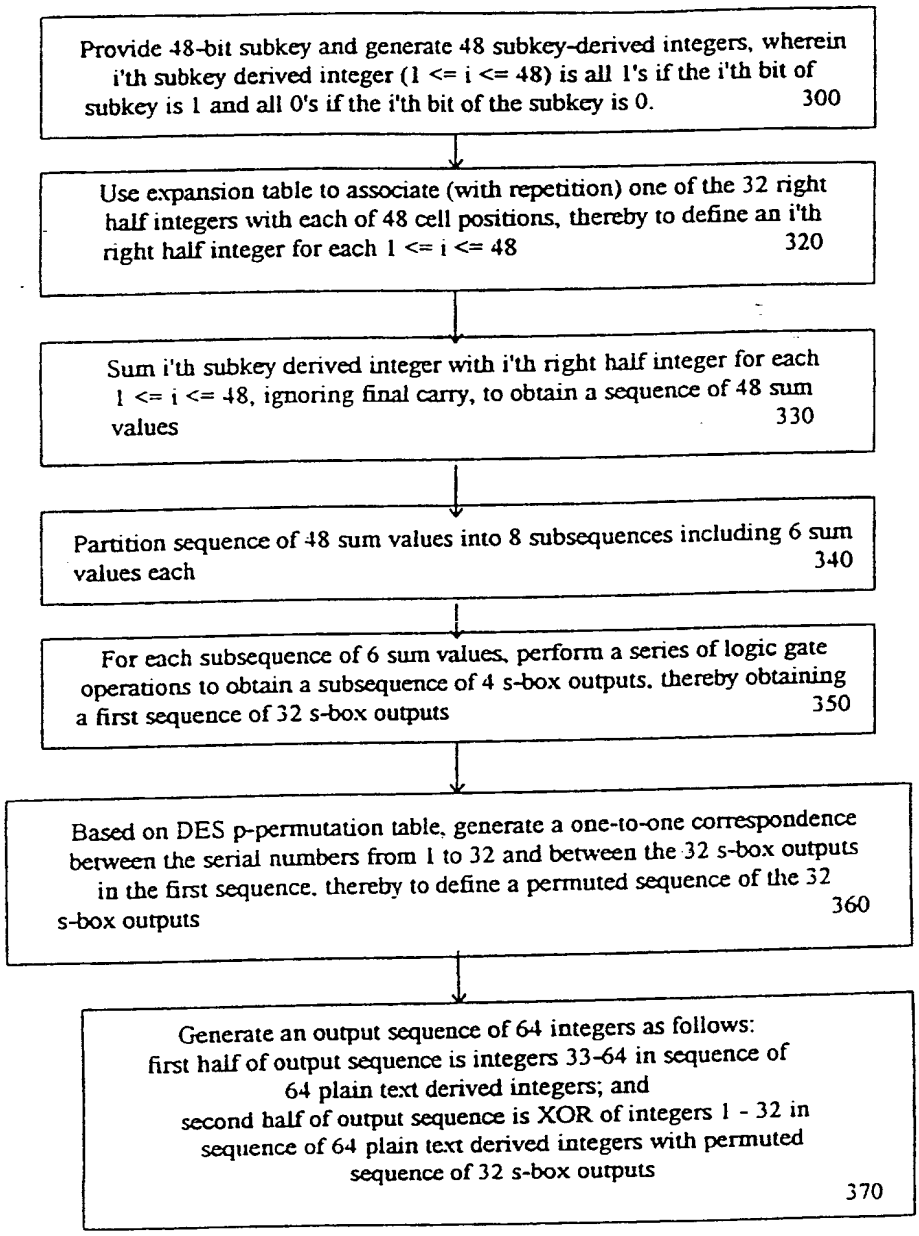


Figure 30

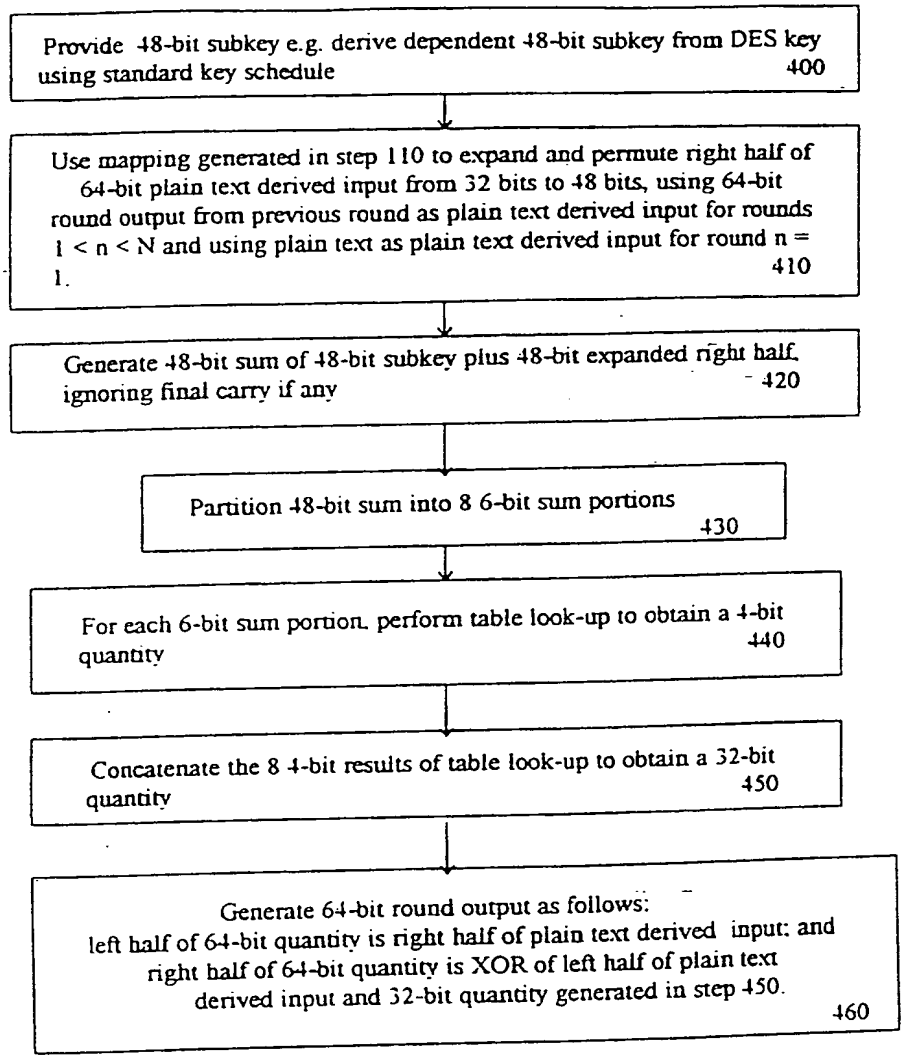


Figure 31

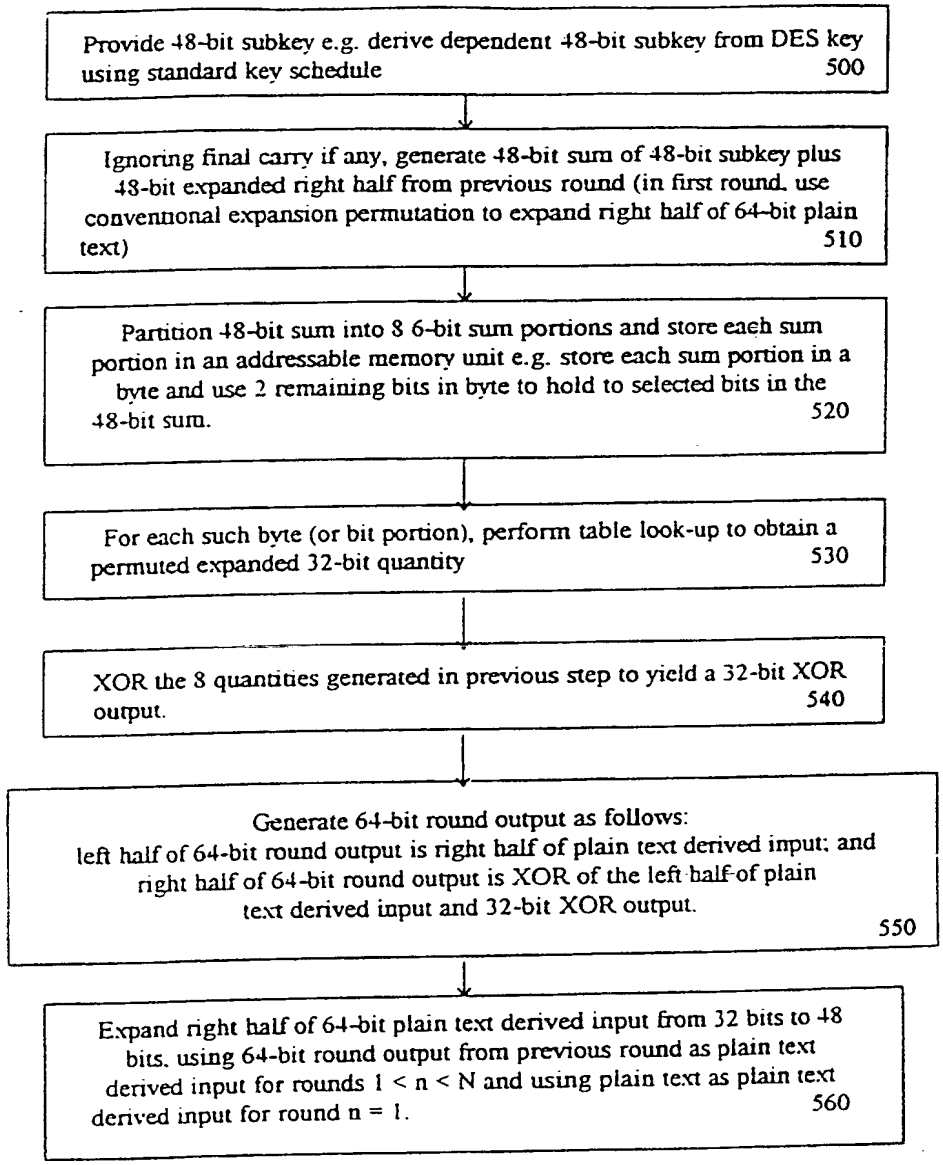


Figure 32

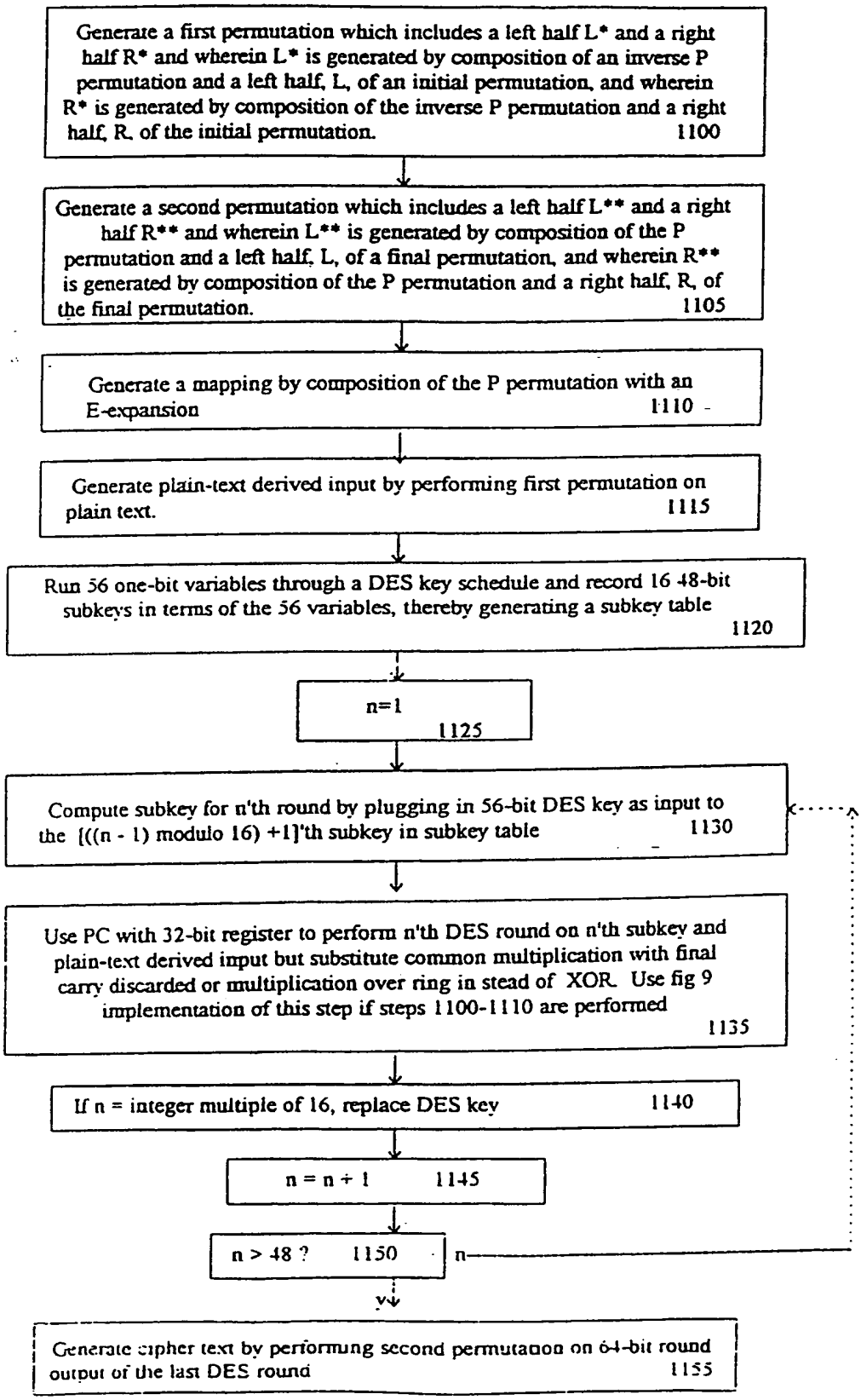


Figure 33

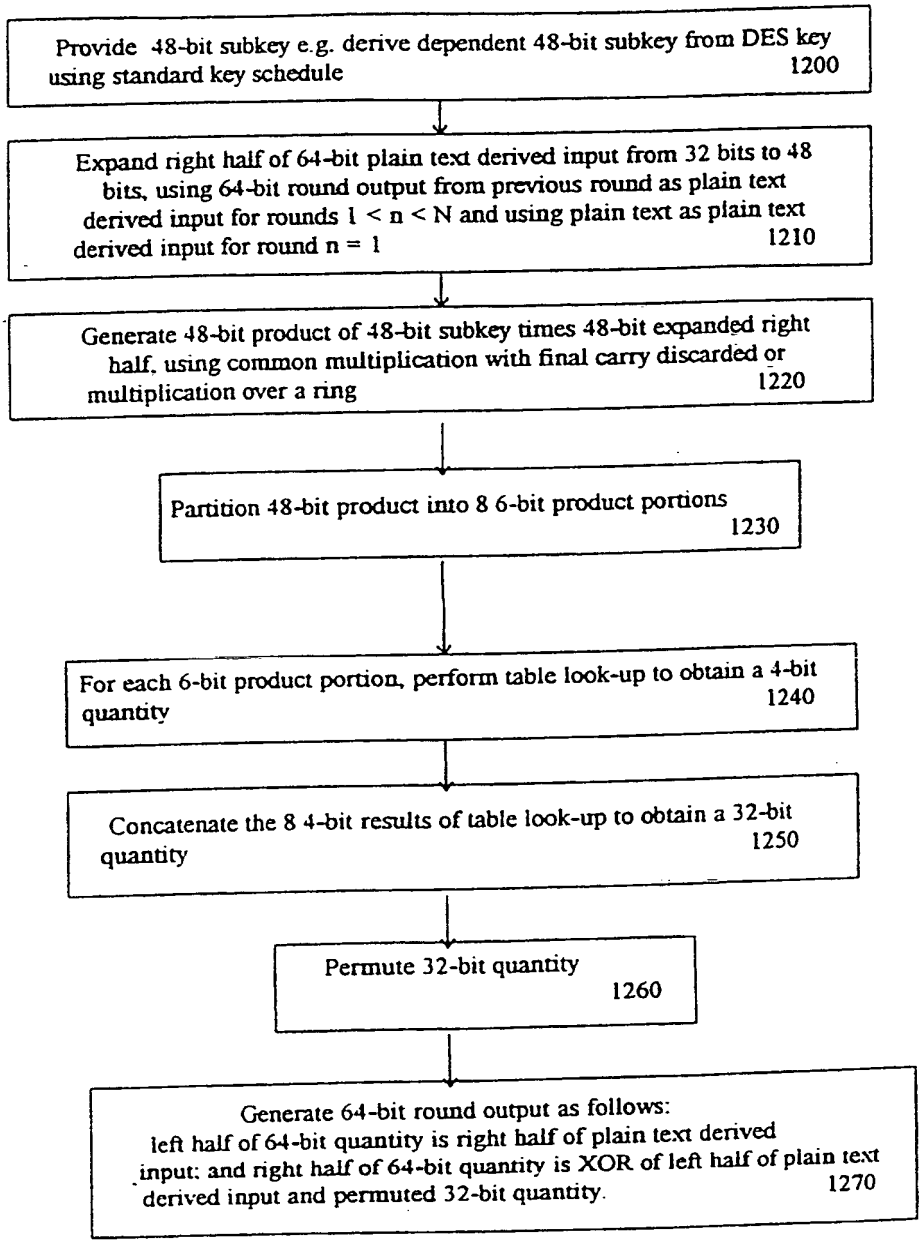


Figure 34

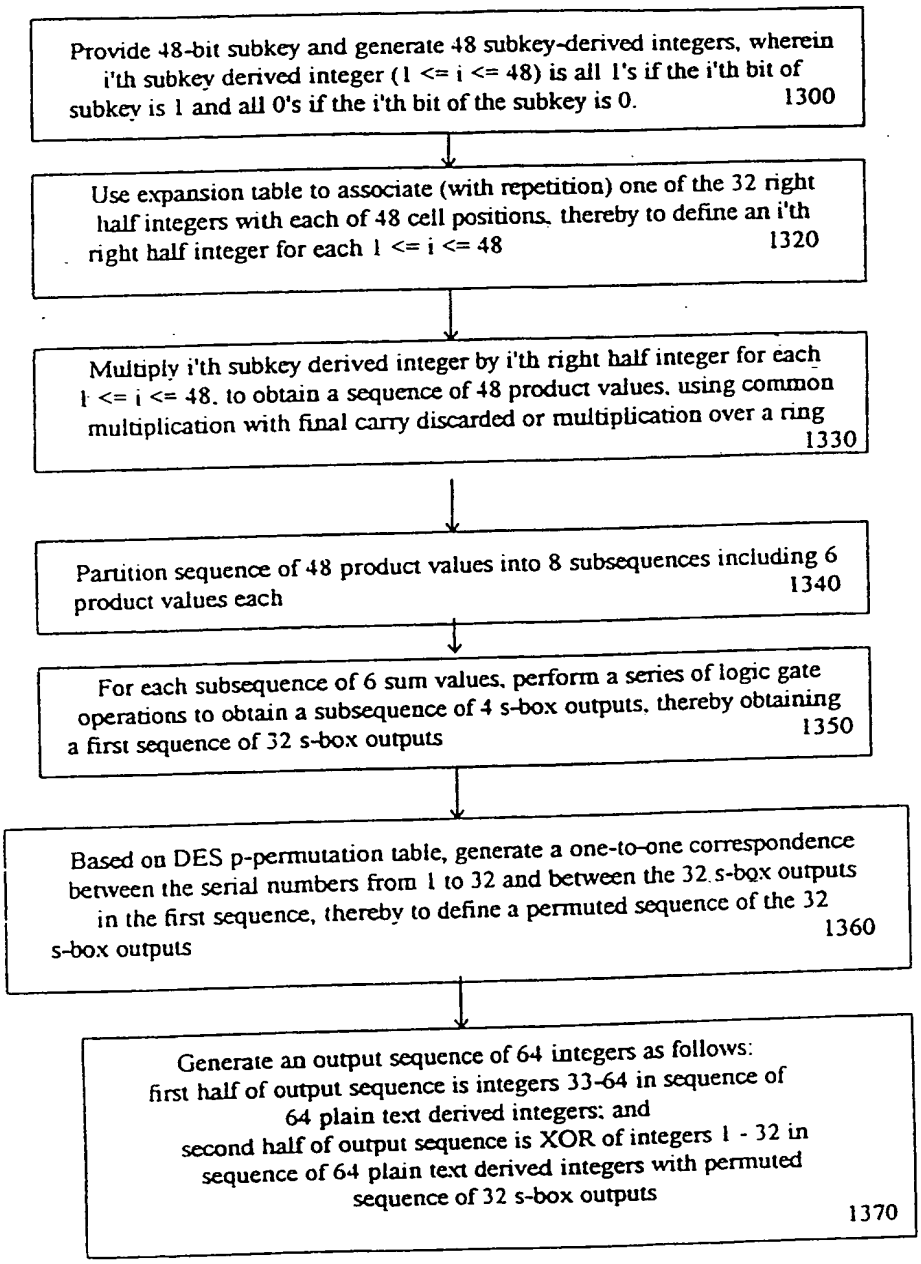


Figure 35

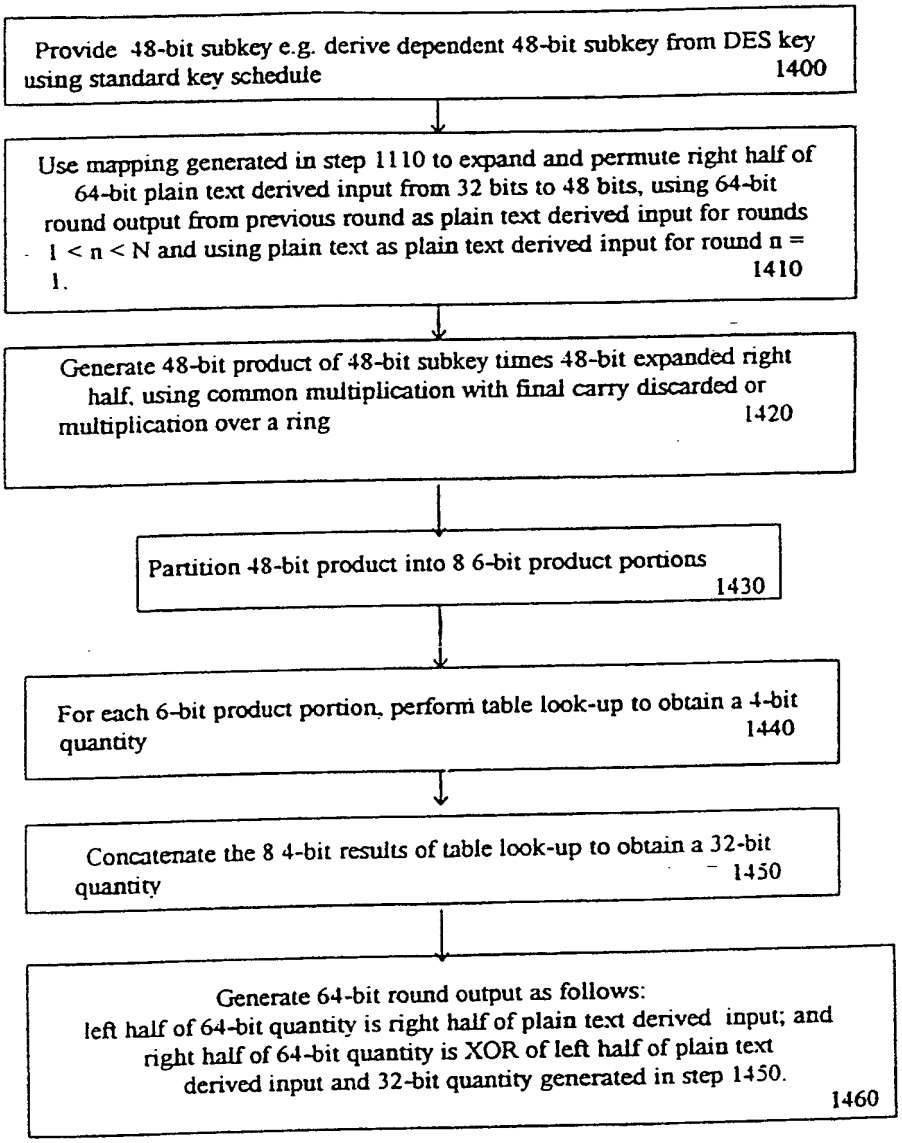
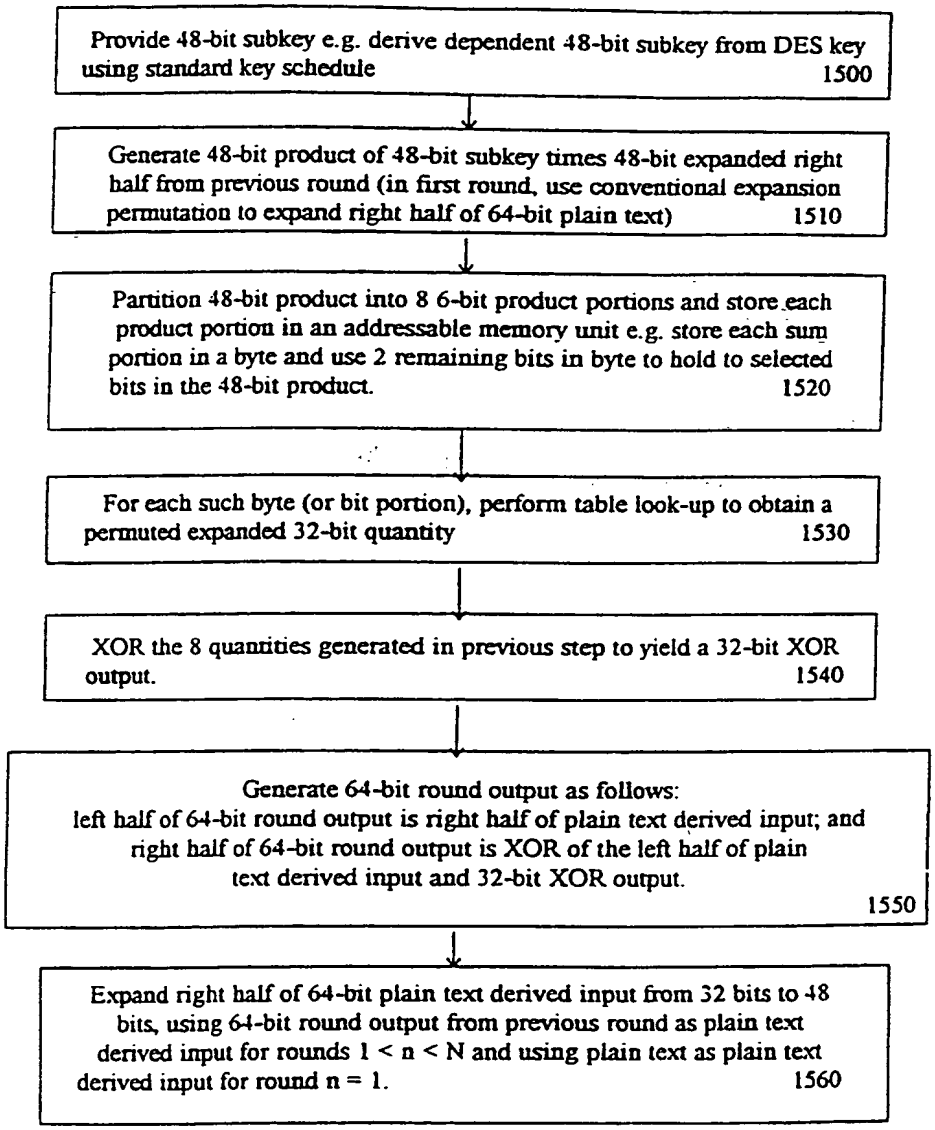


Figure 36



**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : **H04L 9/28** | **A3**

(11) International Publication Number: **WO 99/08411**

(43) International Publication Date: 18 February 1999 (18.02.99)

(21) International Application Number: PCT/IL98/00369

(22) International Filing Date: 6 August 1998 (06.08.98)

(30) Priority Data:

| | | |
|---|---|---|
| 121499 | 8 August 1997 (08.08.97) | IL |
| 121500 | 8 August 1997 (08.08.97) | IL |
| 124705 | 1 June 1998 (01.06.98) | IL |

(71)(72) Applicant and Inventor: STIEBEL, Jonathan [US/IL]; Goldberg Street 7/10, 76283 Rechovot (IL).

(74) Agent: FRIEDMAN, Mark, M.; Beit Samueloff, Haomanim Street 7, 67897 Tel Aviv (IL).

(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, HR, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, US, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

**Published**
*With international search report.*

(88) Date of publication of the international search report:
2 November 2000 (02.11.00)

(54) Title: NEW OPERATION FOR KEY INSERTION WITH FOLDING

(57) Abstract

MultiDES based systems with bit–slice implementation, one embodiment of the method of the present invention, is a new cipher based on a modification of bit–slice implementation of DES. Therein, the exclusive–or is replaced within the F function with a form of multiplication. Thus, every simultaneous encryption depends in all of the bits of input into the s–box on every other parallel encryption. Any invertable group operation could be used in place of multiplication. The principle requirement is that every input bit will influence every output bit. The operation need not be easily invertable, for example, common multiplication using exclusive–or to fold the upper and lower halves of the result yields a strong candidate. The method of the present invention uses a careful form of folding so that the inputs to any s–box depend on at least half of the input bits. MultiDES based systems with bit–slice implementation are particularly preferred, one embodiment of the method of the present invention. The recommended key schedule for Feistel and other blocks ciphers uses the block cipher to cause complete mixing of the key bits and pseudo–random expansion into conveniently sized subkeys. A subkey chaining mode for influencing future encryptions of block ciphers in place of cipher block chaining mode is proposed. A Feistel structure allowing for further extension of block length for subkey chaining output is proposed.

*FOR THE PURPOSES OF INFORMATION ONLY*

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **AL** | Albania | **ES** | Spain | **LS** | Lesotho | **SI** | Slovenia |
| **AM** | Armenia | **FI** | Finland | **LT** | Lithuania | **SK** | Slovakia |
| **AT** | Austria | **FR** | France | **LU** | Luxembourg | **SN** | Senegal |
| **AU** | Australia | **GA** | Gabon | **LV** | Latvia | **SZ** | Swaziland |
| **AZ** | Azerbaijan | **GB** | United Kingdom | **MC** | Monaco | **TD** | Chad |
| **BA** | Bosnia and Herzegovina | **GE** | Georgia | **MD** | Republic of Moldova | **TG** | Togo |
| **BB** | Barbados | **GH** | Ghana | **MG** | Madagascar | **TJ** | Tajikistan |
| **BE** | Belgium | **GN** | Guinea | **MK** | The former Yugoslav Republic of Macedonia | **TM** | Turkmenistan |
| **BF** | Burkina Faso | **GR** | Greece | | | **TR** | Turkey |
| **BG** | Bulgaria | **HU** | Hungary | **ML** | Mali | **TT** | Trinidad and Tobago |
| **BJ** | Benin | **IE** | Ireland | **MN** | Mongolia | **UA** | Ukraine |
| **BR** | Brazil | **IL** | Israel | **MR** | Mauritania | **UG** | Uganda |
| **BY** | Belarus | **IS** | Iceland | **MW** | Malawi | **US** | United States of America |
| **CA** | Canada | **IT** | Italy | **MX** | Mexico | **UZ** | Uzbekistan |
| **CF** | Central African Republic | **JP** | Japan | **NE** | Niger | **VN** | Viet Nam |
| **CG** | Congo | **KE** | Kenya | **NL** | Netherlands | **YU** | Yugoslavia |
| **CH** | Switzerland | **KG** | Kyrgyzstan | **NO** | Norway | **ZW** | Zimbabwe |
| **CI** | Côte d'Ivoire | **KP** | Democratic People's Republic of Korea | **NZ** | New Zealand | | |
| **CM** | Cameroon | | | **PL** | Poland | | |
| **CN** | China | **KR** | Republic of Korea | **PT** | Portugal | | |
| **CU** | Cuba | **KZ** | Kazakstan | **RO** | Romania | | |
| **CZ** | Czech Republic | **LC** | Saint Lucia | **RU** | Russian Federation | | |
| **DE** | Germany | **LI** | Liechtenstein | **SD** | Sudan | | |
| **DK** | Denmark | **LK** | Sri Lanka | **SE** | Sweden | | |
| **EE** | Estonia | **LR** | Liberia | **SG** | Singapore | | |

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/IL98/00369

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :H04L 9/28
US CL :380/28, 29;

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 380/28, 29, 37, 42, 44, 4:

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS
terms: folding, Data Encryption Standard,, bit slicing, wdes

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim N |
|---|---|---|
| X,P | US 5,748,741 A (JOHNSON et al.) 05 May 1998 (05.09.98) - Abstract, figure 7 | 1 |
| X<br>--<br>Y | US 5,454,039 A (COPPERSMITH et al.) 26 September 1995 (26.09.95) - entire document. See Abstract | 3<br>--<br>11 |
| X<br>--<br>Y | US 3,962,539 A (EHRSAM et al.) 08 June 1976 (08.06.76) - entire | 2, 4, 9, 63, 64<br>89, 90, 118-123<br>----<br>11, 42, 43, 130 |

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

| | | | |
|---|---|---|---|
| * | Special categories of cited documents | "T" | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| "A" | document defining the general state of the art which is not considered to be of particular relevance | "X" | document of particular relevance, the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| "E" | earlier document published on or after the international filing date | "Y" | document of particular relevance, the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| "L" | document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | "&" | document member of the same patent family |
| "O" | document referring to an oral disclosure, use, exhibition or other means | | |
| "P" | document published prior to the international filing date but later than the priority date claimed | | |

| | |
|---|---|
| Date of the actual completion of the international search<br>19 JANUARY 1999 | Date of mailing of the international search report<br>27 JUN 2000 |
| Name and mailing address of the ISA/US<br>Commissioner of Patents and Trademarks<br>Box PCT<br>Washington, D.C. 20231<br>Facsimile No. (703) 305-3230 | Authorized officer<br>James R. Matthews<br>PINCHUS M. LAUFER<br>Telephone No. (703) 306-4160 |

Form PCT/ISA/210 (second sheet)(July 1992)*

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/IL98/00369

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| X<br>--<br>Y | BIHAM, E. "A Fast New DES Implementaion in Software", Proceedings of the Fast software encryption Workshop, (ublished as Lecture Notes in Computer Science) pages 260-272 January 1997. See in particular section 3.1 | 7-10, 50-58, 60-62, 65. 66-69, 74-81, 86, 88, 91-96<br>----<br>11, 59, 70-73, 100-104, 106-108 |
| X,P<br>----<br>Y,P | US 5,724,428 A (RIVEST) 03 March 1998 (03.03.98) - entire | 5, 17, 26-28, 30-31, 33, 35, 97,99, 113-116<br>---<br>44, 59, 70-73, 100-104, 106-108, |
| X,E<br>----<br>Y,E | US, 5,838,794 A (MITTENTHAL) 17 November 1998 (17.11.98) - entire document | 2-4, 6, 9, 10, 26-28, 33, 35-37, 60, 63, 64, 90, 97, 118-123<br>----<br>130 |
| X<br>--<br>Y | US 5,319,705 A (HALTER et al.) 07 June 1994 (07.06.94) - entire document. In particular, see Abstract and figures 3 and 12-14. | 39, 40, 41, 45, 46-49, 124-127<br>-----<br>42-44, 130 |
| A | US 5,623,549 A (RITTER) 22 April 1997 (22.04.97) - entire document | 1-130 |

Form PCT/ISA/210 (continuation of second sheet)(July 1992)*